UNIMORE

UNIVERSITÀ DEGLI STUDI DI
MODENA E REGGIO EMILIA

Department of Engineering "Enzo Ferrari"
PhD Course in Information and Communication Technologies (ICT)
University of Modena and Reggio Emilia

XXXVIII Cycle

The Architecture of Time

Modeling, Efficiency, and Discovery in Temporal Data

SUPERVISOR

Prof. Francesco Guerra

CANDIDATE

Giacomo Guiduzzi

COURSE COORDINATOR

Prof. Rovati Luigi

Academic year 2024/2025

"Careful. We don't want to learn from this."

— Bill Watterson, "Calvin and Hobbes"

I dedicate this work to all the people I met during my three years of research, in Italy, Spain, or across the world.
To the New Happy Movement, which gave me hope for the future of humanity.
To all the people who supported me, loved me, and sustained me during the hardest of times.

Grazie. Vi voglio bene.

Thank you. I love you.

Gracias. Os quiero.

Danke. Ich liebe euch.

Abstract

The proliferation of Time Series Data (TSD) across diverse domains presents persistent and critical challenges in modern Data Science (DS), demanding scalable, robust methods for forecasting, efficient resource management, and meaningful pattern discovery. The sheer volume and high-dimensionality of contemporary temporal datasets necessitate a comprehensive, architecture-driven approach to analysis.

This thesis addresses these demands through an investigation structured around three central and interconnected pillars: Modeling, Efficiency, and Discovery.

(i) Modeling and Systematic Evaluation. To overcome limitations in comparative performance and application-specific bias, this work establishes a rigorous, systematic benchmarking framework for time series forecasting models and datasets. This framework moves beyond isolated application studies to provide a critical, scalable foundation for evaluating model robustness and predictive performance across diverse temporal structures, thereby promoting methodological generalization.

(ii) Efficiency in Big Data. Addressing the vast resource requirements inherent in temporal Big Data, the research explores the optimization of Compact Data Structures (CDS) for Time Series (TS). It investigates techniques, such as the strategic exploitation of temporal motifs, to achieve highly efficient, queryable data compression. The focus is placed on optimizing the crucial design trade-offs required to manage large-scale data resources while maintaining full analytical capability.

(iii) Discovery in Complex Systems. To advance pattern detection in high-complexity environments, the thesis contributes two distinct discovery methodologies. First, it introduces a novel spatio-temporal clustering technique that effectively incorporates both geographic and time-based information to facilitate the discovery of complex geological phenomena from PS-InSAR satellite data. Second, the theme of discovery extends to relational data with an exploration of Explainable Entity Matching, evaluating the application of Language Models and Large Language Models (LLMs) to enhance transparency and accuracy in data linkage tasks.

Collectively, this research reframes the handling of ubiquitous temporal data not merely as a search for optimized algorithms, but as a comprehensive engineering challenge. The architectural design — from systematic model evaluation frameworks and resource-efficient methodologies to integrated discovery techniques — is shown to be paramount for achieving impactful and scalable solutions across diverse data science disciplines.

Table of Contents

List of Figures

List of Tables

List of Definitions

List of Abbreviations

APS:	**A**tmospheric **P**hase **S**creen
BSS:	**B**lind **S**ource **S**eparation
CDS:	**C**ompact **D**ata **S**tructures
DL:	**D**eep **L**earning
EEM:	**E**xplainable **E**ntity **M**atching
EM:	**E**ntity **M**atching
HPC:	**H**igh **P**erformance **C**omputing
ICA:	**I**ndependent **C**omponent **A**nalysis
IoT:	**I**nternet **o**f **T**hings
IWT:	**I**nteger **W**avelet **T**ree
MICE:	**M**ultivariate **I**mputation by **C**hained **E**quations
ML:	**M**achine **L**earning
PAA:	**P**iecewise **A**ggregate **A**pproximation
PCA:	**P**rincipal **C**omponent **A**nalysis
PS-InSAR:	**P**ersistent **S**catterer Interferometry
RAG:	**R**etrieval-**A**ugmented **G**eneration
RLZ:	**R**elative **L**empel-**Z**iv
SAX:	**S**ymbolic **A**ggregate appro**X**imation
SOTA:	**S**tate **O**f **T**he **A**rt
ST:	**S**patio-**T**emporal
TS:	**T**ime **S**eries
TSA:	**T**ime **S**eries **A**nalysis

TSD: **T**ime **S**eries **D**ata

TSF: **T**ime **S**eries **F**orecasting

CHAPTER 1

Introduction

The ability to model, predict, and efficiently manage temporal data has become a defining characteristic of advanced data science and engineering in the twenty-first century. From understanding subtle seismic shifts in the Earth's crust to forecasting the complexities of human social systems, time series data provides an essential dimension. However, the sheer volume, velocity, and heterogeneity of modern data streams — often termed Big Data — pose significant challenges to traditional analytical methods, demanding novel solutions in both model design and data infrastructure [SKG17, Yu21].

This thesis presents a systematic investigation into the challenges of designing adaptable architectures for complex temporal data. Rather than focusing on a single domain or a monolithic algorithm, the research explores the necessary frameworks for Modeling, Efficiency, and Discovery across diverse application areas. The work is motivated by real-world problems requiring the integration of multi-modal data and constrained by the practical necessities of scalable data handling.

The following chapter establishes the necessary context for this work, beginning with a detailed discussion on the scope of temporal data applications, defining the current research gaps, and clearly articulating the goals and major contributions that comprise this doctoral research.

1.1 The Challenge of Temporal Data's Broad Context

In the physical and computational sciences, temporal dependency is a fundamental characteristic of nearly all measured phenomena. Even high-frequency events, while appearing instantaneous at macro scales, possess an underlying temporal structure that can be captured and analyzed. Given a defined period, systematic measurement of a phenomenon enables the observation of inherent dynamics, such as **trends**, **periodicity**, and **seasonality**, facilitating the deduction of its behavior and the influence of external factors. Specifically, a **T**ime **S**eries (TS) is formally defined as a set of sequentially ordered data points, typically on the same scale, indexed by a time-like parameter [Bri15]. The time parameter may range over integers or real numbers, with **T**ime **S**eries **D**ata (TSD) referring to a segment of the overall series.

During the twentieth century, fundamental discoveries were made to the field of **T**ime **S**eries **A**nalysis (TSA). Thanks to the ubiquity of TS across fields, different ways to represent TS were formalized [Bri15]. Time Series take on a dazzling variety of shapes and forms, depending on the phenomenon they represent. TSA is a field fundamentally rooted in statistical concepts, seeking to model implicit patterns, probabilities, and distributions [Bri15]. The classical methods of TSA rely on decomposition models, where the observed signal is broken down into constituent components, such as the sum of a signal, or mean function, and residual noise (series = signal + noise). Other decompositions include the fundamental forms: series = trend + seasonal + noise [NIS03] or approximating the series as a sum of cosines. These models define a trend as **a slowly evolving change**, and a seasonal component as **a cyclic movement with a fixed period**.

Classical TSA founded on statistical models such as the **Box-Jenkins** methodology and **Exponential Smoothing** provided reliable tools for low-volume data [Wil16]. However, the subsequent increase in data volume and complexity presents significant scaling challenges for these methods in large-scale, high-velocity environments, necessitating the exploration of machine learning and dedicated data architectures.

The transition from classical, manually analyzed time series — such as those encountered in quality control or single-sensor monitoring — to modern data generation has introduced a monumental shift in scale. The ubiquitous deployment of **I**nternet **o**f **T**hings (IoT) devices, sensor networks, web logs, and high-frequency financial trading systems now generates massive streams of TSD defining the **Big Data** paradigm [SKG17, WYY15, Yu21]. This deluge emphasizes the Volume (sheer quantity) and Velocity (speed of generation) challenges, overwhelming traditional TSA tools designed for smaller datasets. Consequently, the research focus has broadened throughout the years from statistical modeling to include efficient data architectures capable of storing, querying, and processing these massive streams with minimal resource cost, directly impacting the ability to perform real-time analysis [KBY17, TSM18, Yu21].

Beyond sheer scale, modern temporal data presents a challenge in variety and complexity. TS can be **univariate** (single variable, e.g., stock price) or **multivariate** (multiple variables, e.g., a multi-sensor array), **regularly sampled** (fixed interval) or **irregularly sampled** (variable interval) [Ben+25, NIS03]. More critically, the problem semantics are vastly different across domains, requiring highly specialized methodologies: models for geospatial monitoring (detecting geological shifts) differ fundamentally from those for judicial prediction (forecasting social outcomes) or music genre recognition [ANA22, Mas+25]. Furthermore, many high-impact problems are multi-modal, requiring the fusion of TSD with other data forms, such as textual features (NLP), relational information (Knowledge Graphs), and spatial coordinates [Bar+23, Pag+24].

In synthesis, the complexity of modern TSD — characterized by its scale, speed, heterogeneity, and the need for multi-modal integration — renders simple, off-the-shelf analytical solutions inadequate. This necessitates a move toward systematic, architecture-focused research. There is a critical lag in the development of robust, consensus-driven benchmarking frameworks to assess model reliability and a clear deficiency in scalable, query-efficient data structures designed specifically for temporal big data [CS23]. Addressing these limitations is paramount to translating the immense potential of TSD into reliable, impactful applications. Section 1.2 articulates the primary gaps in the current body of work that this thesis aims to address.

1.2 The Research Gap: A Need for Systematic and Scalable Temporal Architectures

Despite the foundational importance of TSD and continuous innovation in ML, the field is currently grappling with three interconnected challenges that hinder the translation of theoretical advances into reliable, high-impact systems. This thesis is specifically designed to address these gaps by developing and investigating new systematic and architectural frameworks for temporal data.

Gap in Modeling: Lack of Systematic Benchmarking and Reliability. The exponential growth in TS forecasting methodologies — spanning classical statistical models, specialized **M**achine **L**earning (ML) techniques, and **D**eep **L**earning (DL) architectures — has led to a state of fragmented and often non-reproducible evaluation. Current literature suffers from several key shortcomings:

1. Models are frequently evaluated on bespoke, domain-specific datasets with inconsistent performance metrics [CTM20, HAB22], making objective, cross-study comparisons extremely difficult.
2. There is no universally adopted, dynamic, and systematic framework capable of assessing what models perform best across a heterogeneous landscape of TSD characteristics [Mey+25, Qiu+24, Shc+25] (e.g., length, frequency, trend strength, noise).
3. The emphasis on incremental novelty in algorithm design often overshadows the critical need for a robust, systematic understanding of model reliability and failure modes under varying real-world conditions [Gol+24, Li+23a, Zho+25a]. This gap necessitates the creation of a systematic benchmarking architecture to provide the field with a reliable and transparent foundation for future model development and selection.

Gap in Efficiency: Inadequate Architectures for High-Velocity Data. The challenges of Volume and Velocity inherent to the Big Data paradigm place severe demands on data infrastructure [KBY17, SKG17]. While numerous general-purpose data storage and compression techniques exist, they are often sub-optimal for TSD:

1. Storage vs. Query Trade-off: Many highly effective compression schemes prioritize space reduction, often requiring resource-intensive full decompression to support analytical operations, searches, or queries. This defeats the purpose of efficiency in high-velocity environments [CS23, LPK22].
2. Lack of Temporal Awareness: Standard storage solutions typically do not exploit the inherent temporal dependencies and redundancies within the data, leading to suboptimal compression ratios for time-series specific workloads [CS23, Gue+25].

Gap in Discovery: Difficulty Bridging Complex Multi-Modal Applications. Successful application of TSD in complex, high-impact domains requires moving beyond simple forecasting and necessitates discovery and synthesis across disparate data types. However, generalized frameworks to achieve this are lacking:

1. Methodologies designed for one domain, such as geospatial clustering in remote sensing, rarely translate efficiently to others, such as judicial prediction in social systems, highlighting a lack of generalized architectural principles for problem decomposition [CSW23, Mas+25].
2. High-value applications, like forecasting trial duration or robust Entity Matching, depend on the effective fusion of TSD with non-temporal data, including unstructured text (NLP), relational graphs, and spatial metadata [Bar+23, Cha+25, Pag+24]. The architectural principles for intelligently integrating these diverse data streams remain

underdeveloped. This research gap calls for an investigation into bespoke, architecturally informed methods that can effectively leverage temporal, spatial, and semantic information to enable novel pattern discovery and accurate prediction in complex, multi-modal environments.

1.3 Goals and Research Questions

Based on the identified gaps in the systematic evaluation of temporal models, the need for scalable data architectures, and the difficulty in bridging complex, multi-modal applications (Section 1.2), the overarching goal of this doctoral research is to develop and investigate the proposed architectural frameworks for efficient TSD analysis. This goal is pursued through three distinct, yet interconnected, research objectives:

1. **Advancing Systematic Modeling and Evaluation**. The first objective is to move beyond fragmented model evaluation by establishing systematic frameworks that assess the reliability and performance of **TS** methodologies in diverse, complex contexts. The following research questions guide this goal:

    - How can the integration of multi-modal data streams — including temporal sequences, unstructured text (**NLP**), and relational information — be architected to inform and enhance the modeling and prediction of complex societal phenomena (e.g., judicial outcomes)? [ANA22, CSW23]
    - What systematic and comprehensive framework is required to robustly evaluate and benchmark the performance of competing time series forecasting models across a heterogeneous landscape of TSD, thereby providing clarity on model suitability and failure modes? [HAB22, Mey+25]

2. **Engineering Efficient Temporal Data Architectures**. The second objective addresses the critical efficiency gap in Big Data environments by focusing on architectural solutions for data handling that prioritize both space conservation and query performance. The research question related to this goal is:

    - How can Compact Data Structures be effectively engineered to enable highly efficient, query-able compression of massive time series datasets, ensuring that analytical operations and searches can be performed directly on the compressed representation without requiring full decompression? [CS23, Gue+25]

3. **Enabling Discovery through Multi-Dimensional Integration**. The third objective focuses on leveraging new architectural principles to extract valuable insights and patterns from **TSD** when integrated with other critical data dimensions, specifically spatial and semantic information.The final research question is:

- How can dedicated temporal and spatial architectures (e.g., through advanced clustering techniques) be combined to enable novel and reliable pattern discovery in complex geospatial data (e.g., separating geological phenomena in **PS-InSAR** data)? [Gha+24a]

1.4 Thesis Contributions

The research presented in this thesis offers several key contributions to the field of Time Series Analysis and Big Data Architecture. These contributions address the research gaps identified in Section Section 1.2 and provide answers to the research questions posed in Section 1.3.

A Systematic Benchmarking Framework for Time Series Forecasting. To address the lack of reproducibility and standardized evaluation in temporal forecasting, this thesis contributes to the design and implementation of a comprehensive benchmarking environment. This framework enables the rigorous evaluation of diverse time series forecasting algorithms against heterogeneous datasets. The primary contribution lies in the formalization of evaluation protocols and the creation of a modular architecture that allows for the objective comparison of classical statistical models against modern deep learning approaches.

Interpretable Spatio-Temporal Clustering for Geospatial Discovery. Addressing the need for discovery in complex physical systems, this thesis introduces a proposed methodology for the Spatio-Temporal Clustering of satellite interferometry data (PS-InSAR). This contribution enables the automated separation and identification of distinct geological phenomena, such as landslides and subsidence, which are often conflated in traditional analysis [Mas+25]. The developed architecture successfully integrates spatial coordinates with temporal displacement series to reveal latent patterns in ground deformation.

Compact Data Structures for Time Series in Big Data. Addressing the efficiency challenges of large-scale temporal data, this thesis contributes an architectural investigation into Compact Data Structures. This work explores the design and performance trade-offs of the investigated compression schemes – specifically those utilizing Relative Lempel-Ziv (RLZ) and temporal motif discovery. The contribution consists of a detailed analysis of how time-aware data structures can enable queryable compression, significantly reducing storage footprints while maintaining the ability to perform efficient searches without full decompression.

Methodological Proposals for Multi-Modal Legal Prediction. This research provides an exploratory analysis for prediction in the complex judicial domain. By investigating the integration of multi-modal data sources — specifically time series, unstructured legal text (NLP), and image-related features data — this work explores current approaches for an architecture intended to forecast criminal trial durations. This contribution highlights the unique challenges of "social time" and establishes a roadmap for data-driven modeling in high-stakes legal environments. [CSW23]

Advancements in Explainable Entity Matching. Finally, this thesis contributes to the field of data integration through an exploration of Explainable Entity Matching. By leveraging and evaluating Large Language Models (LLMs), this work introduces methods for not only identifying duplicate entities across diverse datasets (such as job postings and CVs) but also providing human-understandable explanations for matching decisions, thereby increasing the transparency of automated data fusion processes. [Bar+23, Pag+24]

1.5 The Architecture of Time: A Layered Abstraction Framework

To investigate the multifaceted nature of information discovery in temporal and relational domains, this thesis adopts a layered abstraction framework. This "architectural through-line" organizes the diverse research contributions — ranging from low-level infrastructure to high-level semantic reasoning — into a cohesive hierarchy of functional layers. As illustrated in Figure 1.1, each layer provides the necessary abstractions upon which the subsequent tier is constructed.

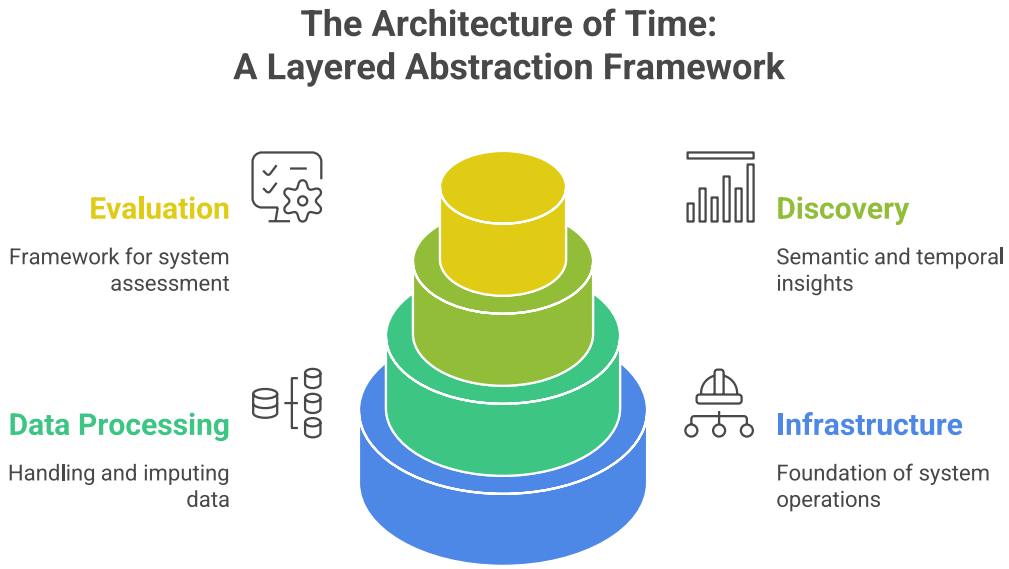


Figure 1.1: Stack diagram that illustrates the abstraction layers each component of a system for time series data processing relies upon, and their role in the system.

The Infrastructure Layer (Physical Representation). The Infrastructure Layer is the foundational tier of the architecture, governing how temporal data is physically represented and retrieved. The primary objective at this level is to resolve the **Storage-Queryability Trade-off**: achieving significant bit savings without sacrificing the ability to interrogate the data in its compressed domain. Chapter 5 instantiates this layer through the development of **Relative Lempel-Ziv** (**RLZ**) algorithms enhanced by **Suffix Array** motif discovery. By moving from stochastic sampling to deterministic, "educated" reference construction, this work explores how to architect time for efficient large-scale storage.

The Data Processing Layer (Signal Integrity). The Data Layer sits above the physical representation and is responsible for transforming raw, incomplete, or noisy inputs into "model-ready" signals. This layer addresses the Integrity Gap – the loss of temporal continuity caused by irregular sampling and sensor failure. Chapter 3 addresses this layer through a systematic evaluation of **Imputation Methodologies**. By benchmarking how simple (e.g., LOCF) and complex deep-learning-based methods (e.g., BRITS, SAITS) reconstruct missing values, we define the protocols necessary to maintain signal fidelity before high-level discovery occurs.

The Discovery Layer (Knowledge Extraction & Mo

deling). The Discovery Layer represents the "intelligence" tier of the architecture spine, where signals are transformed into actionable knowledge. This layer is bifurcated into **Temporal Modeling** and **Semantic Fusion**, addressing the challenge of extracting meaning from both quantitative sequences and unstructured relational data. This layer is instantiated in Chapter 3 via Native ISTS architectures (e.g., PrimeNet) for temporal forecasting and in Chapter 6 through **Explainable Entity Matching** (**EEM**) and **LLM-driven semantic linkage**. It asks: How can we architect systems that not only predict outcomes but also justify the semantic links between disparate entities?

The Evaluation Layer (Methodological Governance). The Evaluation Layer serves as the "vertical spine" that intersects all other tiers. It defines the formal protocols, benchmarks, and metrics required to ensure that discovery is statistically sound and replicable. This layer is formalized in the **ISTS Benchmarking Framework** (Chapter 3), which enforces **Rolling-Origin Validation** and identifies the "**Reliability Gap**" via R^2 quadrant analysis. It provides the necessary governance that prevents the fixed-origin fallacy and ensures that performance gains are not artifacts of specific missingness realizations.

1.6 Scope and Out-of-Scope Discussion

The primary focus of this thesis, "**The Architecture of Time: Modeling, Efficiency, and Discovery in Temporal Data**", is centered on the three core contributions detailed in Section Section 1.4: systematic TS evaluation, efficient data compression architectures, and domain-specific spatio-temporal discovery. However, the necessity of integrating **TSD** with other complex data modalities, first encountered in the judicial prediction task, naturally led to adjacent explorations in data fusion and knowledge extraction. These secondary projects, while important for addressing broader challenges in data architecture and integrity, are positioned outside the core **TS** focus:

- **E**xplainable **E**ntity **M**atching (EEM): The work on EEM and its application leveraging Large Language Models (LLMs) is presented as an exploratory study on semantic data fusion. This work addresses the challenge of creating transparent and verifiable links between disparate data entities, a critical precursor to effective multi-modal modeling (Chapter 6).
- LLMs and **R**etrieval-**A**ugmented **G**eneration (RAG): The investigation into the impact of RAG on the code generation capability of LLMs is included as a discussion of information access architecture. This exploration underscores the need for robust mechanisms to supply **contextual and accurate non-temporal data**, which is essential when

leveraging LLMs for tasks like feature engineering or system prototyping in complex data science projects.

These adjacent projects are presented as supplementary chapters or detailed discussions, acknowledging the real-world complexity of data integration, but they do not constitute core, stand-alone contributions to the advancement of **TS** modeling or efficiency, thus clearly defining the boundaries of this doctoral research.

1.7 Thesis Organization

The remainder of this thesis is structured to guide the reader through the foundational, efficient, and exploratory architectures developed during this research:

Chapter 2 provides a comprehensive Literature Review, covering the evolution of time series analysis from classical statistics to deep learning, and establishing the state-of-the-art in data compression and entity matching.

Chapter 3 details the design and implementation of the Systematic Benchmarking Framework, presenting the results of a large-scale evaluation of forecasting models.

Chapter 4 presents the Spatio-Temporal Clustering methodology, illustrating its application in the discovery of geological phenomena from satellite data.

Chapter 5 describes the investigation into Compact Data Structures, evaluating novel architectures for query-efficient time series compression.

Chapter 6 offers an exploratory discussion on Multi-modal Data Integration, encompassing the work on judicial prediction, Explainable Entity Matching, and the role of RAG in LLM-assisted workflows.

Chapter 7 concludes the thesis by synthesizing the research findings, discussing the limitations of the current work, and proposing future directions for the architecture of temporal data.

CHAPTER 2

Literature Review

2.1 The Evolution of Time Series Analysis

The trajectory of **T**ime **S**eries **F**orecasting (TSF) over the last decade and a half (2010 - 2025) represents one of the most volatile and intellectually rich chapters in the history of data science. This fifteen-year trajectory is not merely a chronicle of increasing computational power but a fundamental epistemological shift in how temporal dynamics are modeled, understood, and predicted. To navigate the dense literature of the past decade, one must first establish a robust theoretical framework. The evolution of TSF is best understood not just chronologically, but through the shifting theoretical frameworks that define the state-of-the-art. Central to this discussion is the tension between universal approximation and domain-specific inductive bias. A critical theoretical undercurrent in TSF research is the application of the "No Free Lunch" (NFL) theorem. Originally formalized by Wolpert and Macready, the NFL theorem postulates that averaged across all possible problem sets, no single optimization algorithm is superior to any other [WM97]. In the context of time series, this implies that a model optimized for high-frequency, chaotic financial data (e.g., stock returns) cannot be expected to perform optimally on low-frequency, seasonal demographic data (e.g., annual birth rates) without inductive bias adaptation [No 12, Bro21, RF23].

The NFL theorem bears profound implications for the "Universal Model" aspirations characterizing the 2020s. Between 2010 and 2015, the forecasting community largely adhered to this principle implicitly, favoring specific statistical models for specific data types. However, the Deep Learning era (2016 – 2023) challenged this view, as researchers pursued "universal" architectures, such as the Transformer, designed to generalize across disparate domains.

For instance, the failure of complex Transformers to outperform simple Linear models on datasets with low semantic density reinforces the necessity of matching inductive biases to the data structure [Zen+22]. If the data generation process is inherently linear or exhibits simple seasonality, complex non-linear models are prone to overfitting, effectively modeling noise rather than the underlying signal [Gol+24, Zen+22]. This phenomenon illustrates a direct manifestation of the NFL theorem: the "cost" of a complex model's flexibility is often a degradation of generalization performance on simpler problems.

In TSF, the NFL theorem manifests uniquely due to non-stationarity. Unlike static classification tasks, the underlying distribution of a time series changes over time. A model that performs well on a training set may experience significant performance degradation on a test set if a regime change occurs [VW24]. This phenomenon is often mistaken for inherent model superiority or inferiority rather than a fundamental property of the optimization landscape. As noted in recent critiques of the M-competitions, the "winning" model is often the one most robust to the specific regime shift occurring in the test set, rather than the model that has learned the "true" underlying physics of the system.

2.1.1 The Statistical Era: Local Modeling and Parsimony

Historically, the field was dominated by parametric models such as ARIMA and Exponential Smoothing (ETS). These "local" methods were designed to be fitted per-series, prioritizing parsimony and respecting specific stationarity properties. While highly effective for isolated, low-frequency data, they suffered from a "local modeling" limitation, preventing the sharing of information across related series, such as retail items influenced by common global trends. Early attempts to apply ML using Support Vector Machines (SVMs) or Multi-Layer Perceptrons (MLPs) generally failed to outcome statistical baselines. These "First Wave" ML models struggled because they were often applied in a local context (one MLP per series) on short datasets, leading to overfitted. They lacked the inductive biases for seasonality and trend that were intrinsic to ARIMA and ETS. Furthermore, the feature engineering — such as lag creation and rolling statistics — was laborious and prone to look-ahead bias [SRP24]. It wasn't until the shift from "Local ML" to "Global ML" — training a single model on thousands of series — that significant progress was realized, satisfying the data requirements of deep nerual networks.

2.1.2 The Deep Learning Disruption: From Local to Global

The pivotal moment for DL in forecasting arrived with the realization that neural networks should be trained globally rather than locally. By training a single model on thousands of time series simultaneously, networks could learn universal temporal patterns, effectively "borrowing strength" from disparate data to forecast sparse series. This period, termed the "Temporal Renaissance" (2010 – 2025), witnessed a shift towards Global Forecasting Models (GFMs) [Kon+25]. Unlike statistical models, DL architectures like **DeepAR** and **N-BEATS** allowed for simultaneous training on large datasets, enabling the exploitation of cross-series information [Ore+20, SFG19].

This era also marked a transition toward probabilistic forecasting, moving away from point estimates to provide robust uncertainty quantification. Amazon's DeepAR (2017) represented a significant from point forecasting [SFG19]. Built on Long Short-Term Memory (LSTM) networks, DeepAR was an autoregressive recurrent network that estimated the parameters of a probability distribution (e.g., Negative Binomial for counts, Gaussian for continuous data) rather than a single value. This facilitated robust uncertainty estimation, which is crucial for supply chain optimization where the cost of under-stocking and over-stocking differ [LZ21]. DeepAR demonstrated that a global model could outperform local statistical models given a large and heterogeneous dataset, and improved on ARIMA by effectively handling "cold start" problems (new items with no history) by learning from similar items. This inaugurated the era of "Global Forecasting Models" (GFMs). The survey by Lim and Zohren highlights this shift, noting that DeepAR's success was less about the LSTM architecture itself and more about the global probabilistic formulation [LZ21].

During the M4 Competition (2018), which challenged the community to outperform statistical benchmarks on 100.000 time series, the expectation of DL dominance was not fully met. The winner was a hybrid model: ES-RNN (Exponential Smoothing – Recurrent Neural Network) [FB20]. This approach combined exponential smoothing equations to de-seasonalize and normalize data "on the fly," which was subsequently fed to an LSTM stack. This architecture allowed the RNN to focus on complex non-linear dynamics while the statistical component handled dominant seasonal and trend components [EY25]. The success of ES-RNN demonstrated that inductive biases were still necessary to guide neural networks, and that "end-to-end" learning

without domain knowledge yielded lower performance on these specific benchmarks compared to carefully engineered hybrid systems.

Following M4, Oreshkin et al. introduced N-BEATS (Neural Basis Expansion Analysis for Time Series) in 2019 [Ore+20]. Unlike ES-RNN, N-BEATS was a pure deep learning architecture comprising a deep stack of fully connected layers with residual links, requiring no time-series-specific feature engineering. Crucially, N-BEATS addressed the "Black Box" criticism of DL by employing a doubly residual topology with interpretable "blocks" devoted to learning trends (monotonic functions) and seasonality (periodic functions). N-BEATS was the first pure DL model to outperform the M4 hybrid winner on several benchmarks, suggesting that with sufficient depth and clever architectural constraints (residual stacks), neural networks could learn TS decomposition from scratch without the crutch of statistical pre-processing. [Ore+20]

2.1.3 The Transformer Era and the "Linear" Counter-Revolution

As the Transformer architecture revolutionized NLP with the attention mechanism, the TSF community rapidly adapted it, leading to a proliferation of "X-former" models between 2020 and 2022 followed by a sharp critical backlash in 2023. A major hurdle for standard Transformers in TS was the quadratic complexity of self-attention ($O(L^2)$) with respect to sequence length L. TS often require long look-back windows to capture seasonalities (e.g., looking back 720 hours to capture monthly patterns in hourly data), making standard attention computationally prohibitive. The **Informer** model addressed this with `ProbSparse` attention, a mechanism that selected only the most dominant queries (based on Kullback-Leibler divergence from a uniform distribution) to compute attention scores, reducing complexity to $O(L \log(L))$ [Zho+21]. Informer also introduced a generative decoder to predict long sequences in a single forward pass, avoiding the error accumulation of step-by-step decoding. Subsequent models like Autoformer [Wu+22] and FEDformer [Zho+22] critiqued the point-wise attention of Informer, arguing that temporal dependencies are better captured in the frequency domain or via decomposition. Autoformer replaced self-attention with an "Auto-Correlation" mechanism, discovering period-based dependencies by comparing sub-series. FEDformer utilized Fourier Transform-based attention to capture global frequency patterns, arguing that sparse attention (like in Informer) loses too much information.

In 2023, a landmark paper titled "Are Transformers Effective for Time Series Forecasting?" by Zeng et al. [Zen+22] shook the community. The publication of this article, where researchers challenged the complexity of Transformer-based models, has been a pivotal moment in the literature. The authors demonstrated that on standard benchmarks (Electricity, Traffic, ETT), simple linear models — specifically DLinear (Decomposition Linear) and NLinear (Normalization Linear), introduced in the paper — consistently outperformed the complex Transformers. The paper posited that the self-attention mechanism, which is permutation-invariant (i.e., treating the sequence order as irrelevant without positional encoding), is fundamentally ill-suited for continuous time series where ordering is the primary signal. While positional encodings help, they are often insufficient to preserve the strict temporal causality required for forecasting. Furthermore, the Transformers were found to struggle with trend extrapolation. A simple linear layer can extrapolate a trend indefinitely; a Transformer, bounded by the range of values seen in its attention window (Softmax outputs), often fails to predict values outside the training distribution (distribution shift). DLinear was embarrassingly simple: it decomposed the series into trend and seasonal components (using moving averages) and applied a single linear layer to each. Its superior performance exposed a "validity crisis" in the field: researchers had been

chasing architectural complexity while neglecting the fundamental characteristics of the data. The Transformers were effectively overfitting to the training noise or failing to capture the trend extrapolation, which a simple linear layer handles trivially. [RKN25, Zen+22] The discovery that simple linear models, as DLinear, often outperformed complex self-attention mechanisms on standard benchmarks exposed a "validity crisis". This correction emphasized that for continuous time series, preserving strict temporal causality is often more important than the semantic complexity of the model architecture.

Amidst this debate, TimesNet offered a novel perspective: instead of treating time series as 1D sequences, it transformed them into 2D tensors based on multiple periods (discovered via Fast Fourier Transform, FFT, operations) [Hua+23, Wu+23]. This allowed the use of 2D convolutional backbones, such as Inception blocks [Sze+14], to capture intra-period and inter-period variations. TimesNet represented a successful integration of computer vision techniques into TSF, arguing that "complex" modeling was viable if the data representation (2D vs 1D) was correct. It demonstrated that Convolutional Neural Networks (CNNs), when adapted correctly, could rival or surpass both Transformers and Linear models on complex, multi-periodic data. The community, driven by the insight that DLinear provided, continued working on that line of research and developed new, improved versions of the model, such as RLinear [Li+23a] and GLinear [RKN25], reiterating the fact that architectural complexity is not always a prerequisite for performance.

2.1.4 Scaling and Foundation Models

The most recent evolutionary leap (2024 – 2025) is the arrival of **Time Series Foundation Models** (TSFMs). Inspired by the success of Large Language Models (LLMs), these models are pre-trained on massive repositories of Time Series data (billions of tokens) and fine-tuned or used zero-shot for downstream tasks [Mey+25]. This represents a return to the "Universal Model" ideal, but backed by massive scale rather than just architectural cleverness. The core challenge for TSFMs is tokenization: how to represent continuous numerical values as discrete tokens for Transformer processing. Different strategies have emerged:

- **TimeGPT** (Nixtla): The first prominent TSFM (late 2023). It is a closed-source model trained on over 100 billion data points. It uses a Transformer-based encoder-decoder but keeps the architectural details proprietary. Benchmarks suggest it outperforms statistical baselines in zero-shot scenarios but struggles against fine-tuned local models. It focuses on identifying common temporal patterns (seasonality, trend) across diverse domains [Mil+24, Ye+25].
- **Chronos** (Amazon): Approaches TSF explicitly as a language modeling problem. It quantizes TS values into a fixed vocabulary of buckets (e.g., 4096 tokens). The time series is converted into a "sentence" of tokens, and a standard LLM (e.g., T5 or Llama) predicts the next token (next bucket). This allows Chronos to leverage off-the-shelf LLM optimizations and pre-trained weights. However, the quantization step inherently loses precision, which can be critical for financial or scientific forecasting [Gru+24, Kot+25].
- **TimesFM** (Google): A decoder-only model pre-trained on Google Trends and Wikipedia pageviews. Unlike Chronos, it uses a patching strategy (similar to Vision Transformers), grouping consecutive time points into vectors. This preserves continuous information

better than quantization. TimesFM demonstrates strong zero-shot performance, particularly on long-horizon forecasts, and avoids the precision loss of bucketing [Das+24].

- **Moirai** (Salesforce): A "Universal Forecasting" model designed to handle multivariate series of varying dimensions. It uses a Masked Encoder architecture (like BERT) rather than a causal decoder (like GPT). Moirai introduces a "Any-variate Attention" mechanism to handle arbitrary numbers of input variables, addressing a key limitation of univariate-only foundation models. It learns a "universal distribution" of Time Series dynamics. [Liu+25a]

Current evaluations (2025) indicate a complex reality. While TSFMs like TimesFM and Moirai show impressive Zero-Shot capabilities (beating simple baselines like Seasonal Naive without training), they often fail to beat state-of-the-art statistical ensembles, such as optimized LightGBM or Theta models, that are trained specifically on the target data [Kon+25, LZ21, MSA20, SPA23]. The NFL theorem remains relevant: a general-purpose foundation model cannot always fully capture the idiosyncratic dynamics of a specific dataset (e.g., specific sensor noise) as well as a specialized model. However, **In-Context Learning** is emerging as a bridge, with recent work showing that TSFMs can be improved significantly at inference time by providing "prompts" of similar time series (**In-Context Fine-Tuning**), allowing the model to adapt its internal state without weight updates [God+25]. This suggests that the future may lie in "retrieval-augmented forecasting": finding similar historical series to prompt the foundation model.

Table 2.1 summarizes the most important changes in architecture, their era, and the relative bias. This taxonomy reveals a cyclic trend: the field moved from simple (Statistical) to complex (Recurrent) to highly complex (Transformers), and then sharply corrected back to simple (Linear) before expanding again to a massive scale (Foundation Models).

Table 2.2 compares the Time Series Forecasting paradigms highlighted so far, with a focus on features from training to inference. To summarize, the literature review regarding Time Series Analysis shows that statistical models remain the most efficient choice for individual, high-signal time series. DL (specifically Linear-based DL or Transformers) excels in complex, high-dimensional, multivariate regimes. Foundation Models are currently an optimal choice for "cold start" problems or generic forecasting where training resources are unavailable, but they have not yet rendered task-specific training obsolete.

Category	Core Mechanism	Representative Models	Key Era	Primary Inductive Bias
Statistical / Local	Autoregression, Moving Averages, Exponential Smoothing. Models are trained per-series	ARIMA, ETS, Theta, Prophet, TBATS	Pre-2015 (Dominant), 2015-Present (Baseline)	Stationarity, Seasonality, Parsimony
Recurrent / Dilated	Sequential processing with memory states or dilated convolutions	LSTM, DeepAR, WaveNet, TCN, ES-RNN	2016 - 2019	Sequential dependency, Markov property
Transformer / Attention	Self-attention mechanisms, capturing long-range dependencies via query-key-value pairs	Informer, Autoformer, FEDformer, PatchTST, iTransformer	2020 - 2023	Long-range interaction, Permutation invariance (mitigated by positional encoding)
Linear / MLP	Direct linear mapping or simple multi-layer perceptrons, often with decomposition	N-BEATS, DLinear, NLinear, TiDE, TSMixer	2019, 2022 - 2024	Trend extrapolation, decomposition, simplicity
Foundation Models	Large-scale pre-training on massive corpora, zero-shot transfer, tokenization of time	TimeGPT, Chronos, Moirai, TimesFM, Lag-Llama	2024 - 2025	Transfer learning, Scaling laws, "Language" of time

Table 2.1: Phylogenetic Taxonomy of Time Series Models (2010 - 2025).

Feature	Statistical Models (ARIMA, ETS)	Deep Learning (DeepAR, TimesNet)	Foundation Models (TimesFM, Chronos)
Data Requirement	Low (Works on < 50 points)	High (requires thousands of points or series)	Massive for pre-training; Zero for inference
Training Time	Fast (per series)	Slow (requires GPU)	Instant (Zero-shot); Slow (Fine-tuning)
Accuracy (Short Horizon)	High (often SOTA for simple series)	Competitive, but often overkill	Competitive, usually beats naive baselines
Accuracy (Long Horizon)	Poor (errors compound rapidly)	High (Informer, Autoformer excel here)	High (captures global trends well)
Interpretability	High (components are explicit)	Low (N-BEATS is an exception)	Low ("Black Box")
Multivariate Handling	Limited (VAR is fragile)	Excellent (core strength of DL)	Mixed (Chronos is univariate)
Probabilistic Output	Native (prediction intervals)	Native (quantile output)	Supported (via sampling or mixture density)

Table 2.2: Comparative Analysis of Forecasting Paradigms.

2.1.5 Evaluation Methodologies and Benchmarking

The M-competitions (M4, M5, M6) serve as the "Olympic Games" of forecasting, essential for separating academic hype from real-world performance. Key findings include:

- M4 (2018): M4 proved that combining statistical smoothing with RNNs (ES-RNN) was superior to either approach alone [FB20]. It also highlighted the importance of the OWA (Overall Weighted Average) metric, which balanced accuracy (sMAPE) and bias (MASE), penalizing models that were accurate but reckless. M4 showed that pure machine learning methods often failed to beat a simple combination of statistical methods (Comb), reinforcing the "Ensemble" principle.
- M5 (2020): Highlighted the dominance of Gradient Boosted Decision Trees (LightGBM) over pure DL in hierarchical retail tasks [MSA22]. The M5 competition focused on hierarchical retail sales (Walmart data). The results were a resounding victory for Gradient Boosted Decision Trees (GBDT), specifically LightGBM. The winning solution by Makridakis et al. did not use DL architectures like Transformers or RNNs. Instead, they used LightGBM with extensive feature engineering (lag features, rolling windows) [MSA20]. In tabular data environments (like retail sales with price, promotion, and category attributes), tree-based ensembles often outperform DL because they handle heterogeneous features and scale differences better without extensive normalization.

The M5 also highlighted the difficulty of hierarchical reconciliation, which consists of ensuring that the sum of item forecasts equals the store forecast. While sophisticated reconciliation methods (MinT [SP25]) exist, top performers often used simple bottom-up aggregation or trained separate models for each level. The competition debunked the idea that "end-to-end" DL handles hierarchies naturally; explicit handling of the hierarchy was often required. [Spr+24]

- M6 (2022): The M6 competition shifted to financial forecasting (stock returns and risk). The "No Free Lunch" theorem was in full effect: complex ML/DL models largely failed to beat simple benchmarks or the "efficient market" hypothesis (random walk) [Mak+24]. The competition demonstrated that in low signal-to-noise ratio domains (like finance), architectural complexity often leads to overfitting. The winners often focused on risk management (covariance estimation) rather than pure return forecasting. This highlighted a critical limitation of the TSF evolution: while we have become better at forecasting "predictable" patterns (seasons, trends), we have made little progress in forecasting "unpredictable" stochastic noise [VW24].

2.1.6 The Reproducibility Crisis

Parallel to the model evolution, a crisis of confidence emerged regarding published results. The chase for **S**tate **O**f **T**he **A**rt (SOTA) performance led to suboptimal scientific practices, extensively documented in recent survey papers (2024 – 2025). A widespread issue in TSF papers is **Data Leakage**, which renders benchmarks invalid [AA25, KN22]. Leakages are categorizable into three specific types found in the literature:

- **Temporal Leakage** (Preprocessing): This is the most common and insidious error. It occurs when global statistics (mean, variance) are calculated over the entire dataset (train + test) to normalize the inputs. This leaks information about the future distribution into the training set. For instance, if the test set contains a massive spike, the global variance increases, scaling down the training inputs in a way that "prepares" the model for the spike.
- **Look-ahead Bias** (Feature Engineering): In rolling window evaluations, improper indexing where the target value y_t is accidentally included in the input features X_t. This often happens when creating "lag" features without careful offset management.
- **Test Set Leakage** (Model Selection): Tuning hyperparameters (learning rate, number of layers) on the test set rather than a validation set. This is particularly prevalent in DL papers, where "best epoch" selection is based on test loss. A rigorous evaluation requires a held-out test set that is touched only once.

Furthermore, the field has relied heavily on a small set of datasets (ETT, Electricity, Exchange Rate, Weather). Recent studies argue that these datasets are insufficient for evaluating models: they are too small, lack diversity, and suffer from "benchmark hacking" where models are hyper-tuned to specific dataset artifacts [Kim+25, Mey+25]. The ETT (Electricity Transformer Temperature) dataset, for example, is so widely used that models are now arguably overfitting to the specific noise patterns of those two years of data in China, with zero guarantee of transferability to other power grids. Additionally, many papers use a fixed forecast origin (splitting

data once into train/test, e.g., the last 20% is test), which is statistically fragile. A model might perform well on the last 20% simply because the regime was stable. The best practice, although more computationally expensive, is **Rolling Origin Evaluation** (or **Time Series Cross-Validation**), which evaluates the model over multiple temporal windows to account for regime shifts. The lack of rolling origin evaluation in many Transformer papers casts doubt on their robustness claims [HAB22, Tas00].

This justifies the need for the heterogeneous benchmarking framework developed in this thesis, as a model optimized for seasonal retail data may fail on chaotic financial or geological data. The No Free Lunch Theorem, applied to Time Series Forecasting, makes the creation of a valid and reproducible benchmark a significant challenge, which is addressed in Chapter 3.

2.2 Spatio-Temporal Pattern Discovery

The decade spanning 2015 to 2025 represents a transformative era in the field of Data Science, characterized specifically by the maturation of **S**patio-**T**emporal (ST) pattern discovery [AKK18]. While the preceding decade focused predominantly on managing the magnitude of "Big Data", the current era has redirected its focus toward the complexity of "High-Dimensional Data" – data that is not only voluminous but also intricate, heterogeneous, and intrinsically entangled with the dimensions of space and time [Yan+20a].

The proliferation of sensing technologies — from the constellations of Synthetic Aperture Radar (SAR) satellites monitoring the Earth's crust to the ubiquitous GPS transponders tracking global logistics and the digital breadcrumbs of social interaction — has established a digital ecosystem where location and time are no longer merely metadata tags [AKK18, Yan+20a]. Instead, they have become the primary axes of analysis. This shift has necessitated a fundamental reimagining of unsupervised learning algorithms. Traditional methods, designed for static snapshots of independent and identically distributed (i.i.d.) data, often struggle to capture the dynamic, autocorrelated, and non-stationary nature of ST streams [SP19].

2.2.1 Spatio-Temporal Clustering Methodologies: The Convergence of Space and Time

Clustering, the unsupervised categorization of objects into groups based on similarity, constitutes the bedrock of pattern discovery. In the ST domain, "similarity" is a fluid concept: two objects may be spatially proximate but temporally distant, or they may share a similar trajectory while being geographically separated [Kis+]. The challenge for the 2015 – 2025 period has been to formalize these relationships into mathematical frameworks that are robust to noise and scalable to high dimensions.

Density-Based Spatial Clustering of Applications with Noise (DBSCAN) has long been favored for its ability to discover clusters of arbitrary shape and its inherent outlier detection [Dua+07]. However, its application to TS data requires a nuanced handling of the temporal dimension to avoid conflating disjoint events. The adaptation of DBSCAN into ST-DBSCAN represents a critical methodological advancement [Kis+, SP19]. The core innovation lies in the decoupling of the neighborhood definition. A standard epsilon (ε) neighborhood in Euclidean space is insufficient because spatial proximity does not imply temporal relevance [LDR20].

ST-DBSCAN introduces two distinct threshold parameters:

- Spatial Epsilon (ε_1): Defines the maximum spatial distance for two points to be considered neighbors.
- Temporal Epsilon (ε_2): Defines the maximum time difference allowed between neighbors.

A point is identified as a "core point" — the seed of a cluster — only if its neighborhood, defined by the intersection of these two thresholds, contains a minimum number of points (`MinPts`). This dual-constraint mechanism allows the algorithm to identify "moving clusters", such as a flock of birds or a convoy of vehicles, which maintain a dense formation over time even as their absolute geographic coordinates change continuously [SP19]. Comparative studies have highlighted that while ST-DBSCAN excels at noise rejection — automatically classifying isolated GPS points as sensor errors — it faces limitations when dealing with multi-scale data. For instance, in animal tracking datasets, an animal may exhibit dense movement behaviors in a feeding ground (small spatial scale) and sparse, rapid movements during migration (large spatial scale). A single global density threshold often fails to capture both behaviors simultaneously [LDR20]. To address the limitation of global parameters, researchers have developed hybrid algorithms, such as the Optimization of Initial Points and Variable-Parameter DBSCAN (OVDBSCAN) [Hua+19]. This algorithm, widely applied in financial time-series clustering but highly relevant to geospatial data, employs an optimization routine to dynamically adjust ε and `MinPts` based on the local density of the dataset. By integrating this variable-density clustering with Support Vector Regression (SVR), the OVDBSCAN framework can not only identify clusters but also build predictive regression models specific to each cluster. This is a crucial development for dealing with non-stationarity, a common feature of ST data where the statistical properties of the system change over time. For example, in traffic prediction, the "density" of cars during rush hour differs fundamentally from late-night traffic; OVDBCSAN adapts to these regimes automatically, improving prediction accuracy significantly compared to static-parameter counterparts [Hua+19].

While density-based methods are powerful, their computational cost (often $O(N^2)$) can be prohibitive for massive datasets. Partitioning methods like K-Means are faster ($O(N)$) but traditionally struggle with the "curse of dimensionality" and the assumption of spherical clusters. The Spatiotemporal K-Means (STkM) algorithm addresses these deficits by redefining the objective function [Dor+24]. Instead of treating time simply as an additional dimension in a feature vector — which leads to scaling issues due to unit mismatches between meters and seconds — STkM optimizes a cost function that explicitly accounts for the continuity of time. The algorithm is designed to discover dynamic clusters with static identities. Consider a group of animals moving together: the "cluster" (the group) moves through space-time, but the identity of the group remains constant. STkM minimizes the intra-cluster variance across both spatial and temporal dimensions simultaneously. Theoretical proofs provided in recent literature demonstrate that STkM is particularly effective in the "low-data limit" [Dor+24]. This capability is vital for emerging applications where historical data is sparse. In benchmark evaluations using collective animal behavior datasets, STkM outperformed baseline methods (including standard K-Means and Spectral Clustering) by successfully analyzing multi-scale relationships without requiring extensive parameter tuning or post-processing. It provides a mathematically rigorous way to track evolving groups without the heuristic "stitching" of snapshots often required by simpler methods [Dor+24].

A method that offers a distinct philosophical advantage is Hierarchical Clustering: it does not force the analyst to pre-select the number of clusters (as in K-Means) or a density threshold (as in DBSCAN). Instead, it produces a dendrogram — a tree of nested clusters — that reflects the multi-scale nature of the real world [LDR20]. The state-of-the-art in ST hierarchical clustering revolves around the construction of a robust, generalized distance function. Recent surveys highlight a metric that combines spatial, temporal, and thematic attributes into a single scalar value via a weighted linear combination. The distance $D(c_i, c_j)$ between two clusters or observations is formulated as:

$$D(c_i, c_j) = \frac{1}{\sum_{i=1}^{n} w_i} \left(w_1 \frac{D_{m(x_i,x_j)}}{S_D} + w_2 \frac{D_{t(t_i,t_j)}}{S_T} + \sum_{a=3}^{q} w_a \frac{D_{a(a_i,a_j)}}{S_a} \right) \tag{2.1}$$

This formula encapsulates several critical innovations [LDR20]:

- Adaptive Spatial Scaling (S_D): The scaling factor S_D is not a global constant but an adaptive value derived from the dataset's topology. A common heuristic involves setting S_D based on the k-nearest neighbor distances, where k is derived from the natural logarithm of the sample size ($\ln(N)$). This ensures that "proximity" is relative; what is "close" in a sparse rural dataset might be "far" in a dense urban one.
- Temporal Scaling (S_T): This factor normalizes time differences, allowing the algorithm to handle cyclical time. For example, in analyzing animal behavior or urban crime, 11:00 PM and 1:00 AM are temporally close (2 hours apart), even though they exist on different calendar days. The metric uses cosine transformations to map linear time onto a cycle, ensuring that patterns spanning the midnight boundary are correctly clustered.
- Thematic Integration: The inclusion of non-spatiotemporal attributes (D_a) allows for multi-view clustering. For instance, two trajectories might be spatially and temporally identical but represent different activities (e.g., "walking" vs. "waiting"). By weighting thematic attributes, the hierarchy can distinguish these behavioral modes.

The choice of linkage criteria — the rule used to merge clusters — has a profound impact on the result. A variety of methods are available depending on the specific issue at hand, among which the most frequently utilized are [LDR20]:

- Ward's Linkage: Minimizes the total within-cluster variance. It is highly effective for finding compact, spherical clusters and has shown the highest performance metrics (Homogeneity Score ≈ 0.93) in trajectory simulations.
- Average Linkage (UPGMA): Computes the average distance between all pairs of points in two clusters. It is considered robust to noise and effective for elongated shapes often found in road-constrained movement.

To validate these clusters, the Cluster Over-Time Stability Evaluation (CLOSE) framework was introduced. CLOSE assesses clustering quality by measuring the temporal stability of a time series with respect to its cluster neighbors. A high-quality cluster should not only be cohesive in the feature space but also stable over time; objects should not flip-flop between clusters randomly. This metric provides a "parameter-free" toolkit for selecting the optimal clustering configuration, addressing a major gap in unsupervised ST learning [KTC22].

As data dimensionality increases — for example, in video streams or hyperspectral satellite imagery — traditional distance metrics like Euclidean distance lose their discriminative power. This has led to the adoption of Deep Temporal Clustering (DTC) and Deep Embedded Clustering (DEC). These architectures employ **autoencoders** (often bidirectional LSTMs or Convolutional Neural Networks) to map high-dimensional raw data into a lower-dimensional latent space. The innovation lies in the objective function: rather than separating dimensionality reduction and clustering into two steps, these models optimize a joint loss function containing both reconstruction loss (ensuring the latent code captures the data's essence) and clustering loss (forcing the latent codes to group into tight clusters, often using Kullback-Leibler divergence) [Far+23]. Another example of DL applied to the clustering of ST data is Clustering-Augmented Segmentation (CAS): in remote sensing scenarios, CAS uses a U-Net architecture to perform pixel-level clustering for land cover classification. By preserving spatial details through the U-Net's skip connections while optimizing for cluster compactness, CAS avoids the "blurring" effect common in pure dimensionality reduction techniques [Far+23].

2.2.2 Discovery in Geospatial Data (InSAR)

Persistent **S**catterer Interferometry (PS-InSAR) has revolutionized geodesy and geohazards monitoring. By analyzing the phase difference of radar waves reflected from stable targets (buildings, rocks) over time, scientists can measure ground deformation with millimeter-level precision [FPR00, FPR01]. However, these measurements are notoriously noisy. The radar signal travels through the turbulent atmosphere, accumulating delays caused by water vapor, pressure, and temperature changes. Separating the "signal" (geological deformation) from the "noise" (atmosphere) is the central challenge of modern InSAR processing. An **InSAR Time Series** for a given pixel (φ_{total}) can be mathematically decomposed as:

$$\varphi_{\text{total}} = \varphi_{\text{deformation}} + \varphi_{\text{atmosphere}} + \varphi_{\text{topography}} + \varphi_{\text{orbit}} + \varphi_{\text{noise}} \tag{2.2}$$

The difficulty arises because of **A**tmospheric **P**hase **S**creen (APS), which is spatially correlated (clouds cover large areas) *but* temporally uncorrelated (weather changes daily), and **deformation**, which is often spatially correlated (a subsiding basin) *and* temporally correlated (subsidence is a continuous process). Traditional methods rely on external weather models (like ERA5) to estimate $\varphi_{\text{atmosphere}}$. However, these models often lack the spatial resolution to capture small-scale turbulence, leaving significant residual noise that can obscure subtle geological phenomena [Far+24, Gho+25]. To overcome the limitations of external models, researchers have turned to **B**lind **S**ource **S**eparation (BSS) techniques, which attempt to unmix the signal based purely on statistical properties.

Independent **C**omponent **A**nalysis (ICA) assumes that the mixed signal is a linear combination of statistically independent non-Gaussian sources. In the context of InSAR, this is a powerful assumption. Geological events, such as the sudden inflation of a volcano or the seasonal acceleration of a landslide, often exhibit non-Gaussian statistical distributions. Atmospheric noise, conversely, often tends toward a Gaussian distribution over time. Research conducted on permafrost highways (e.g., the Gongyu Highway in Tibet) and subsidence zones in Italy has demonstrated that ICA is superior to **P**rincipal **C**omponent **A**nalysis (PCA) for this task [Xin+24]. PCA, which relies on maximizing variance and enforcing orthogonality, often fails to separate atmosphere from deformation if the atmospheric signal is strong (high variance). ICA, by focusing on statistical independence, successfully isolates "accelerating" and "decelerating"

deformation trends from the background seasonal noise, achieving a clearer separation of the physical drivers of ground motion [Far+24, Xin+24].

One of the most significant breakthroughs in the 2015 – 2025 period is the integration of **Variational Autoencoders** (VAEs) with density-based clustering to create a fully unsupervised noise removal pipeline. This method, often referred to as VAE-Clustering, has been integrated into open-source software like MintPy [Gho+25]. The mechanisms of VAE-Clustering consist of:

- Latent Encoding: The InSAR stack is flattened and fed into a VAE. The VAE learns to compress the complex time series into a low-dimensional probabilistic latent space. Unlike a standard autoencoder, the VAE learns a distribution, making it generative.
- Source Generation: The decoder attempts to reconstruct the original signals from the latent codes, effectively generating a set of "independent sources" (S) that make up the mixed signal.
- Clustering via FISHDBC: The recovered sources are then clustered using FISHDBC (a variant of density-based clustering). The critical hypothesis here is that deformation signals cluster together due to their spatial consistency (neighboring pixels move similarly). In contrast, atmospheric noise appears as unclustered outliers in the latent space because turbulence is random and spatially incoherent at fine scales.
- Noise Rejection: Sources identified as "noise" (unclustered points) are discarded.
- Reconstruction: The time series is reconstructed using only the clustered (deformation) sources.

Case studies in Mashhad and Tehran (Iran) and Acapulco (Mexico) have validated this approach: the VAE-Clustering method reduced the velocity standard deviation (a proxy for noise) by approximately 70% to 84% compared to standard ERA5 corrections. This 70% to 84% reduction in velocity standard deviation allowed researchers to detect subtle post-seismic deformation signals that were previously buried under atmospheric artifacts [Gho+25].

For analyzing the spatial patterns of the deformation once the noise is removed, a framework combining UMAP and HDBSCAN has emerged as the state-of-the-art [Ryg+23]. In this novel framework, Uniform Manifold Approximation and Projection (UMAP) is used to reduce the high-dimensional displacement Time Series (which may have hundreds of time steps) into a 2D or 3D embedding. UMAP is preferred over t-SNE because it better preserves the global structure of the data, ensuring that the relative distances between clusters are meaningful. Subsequently, Hierarchical DBSCAN is applied to the UMAP embeddings. HDBSCAN improves upon ST-DBSCAN by being robust to varying densities and automatically selecting the number of clusters. It identifies spatially contiguous areas that are deforming similarly. Once clusters are identified, Piecewise Linear Functions (PWLF) are fitted to the centroids of each cluster. This technique automatically detects breakpoints in the Time Series, which are defined as the moments where the deformation rate changes significantly.

This framework was applied to Sentinel-1 data over the Bandung Basin in Indonesia [Ryg+23]. It successfully identified 12 distinct clusters of subsidence. Crucially, the PWLF analysis revealed that while some areas were subsiding linearly (constant rate), others were accelerating (indicating a worsening condition, likely due to increased groundwater extraction) or decelerating. This level of granular insight is impossible with static velocity maps and highlights the power of temporal clustering in geohazards monitoring [Ryg+23].

Complementing these clustering approaches are semi-automated statistical tools like **PS-Time** [Far+24]. This MATLAB-based tool classifies deformation trends into predefined categories (Linear, Quadratic, Bilinear, Seasonal) using a sequence of statistical tests:

- ANOVA F-Test: Determines if a linear trend is statistically significant.
- BIC (Bayesian Information Criterion): Used to determine if a "Bilinear" model (two lines with a breakpoint) explains the data better than a simple linear model.
- Periodicity Checks: Identifies cyclic fluctuations (seasonality) typically caused by the thermal expansion of buildings or the reversible swelling of clay soils.

This taxonomy allows researchers to map not just the magnitude of deformation, but its nature, providing clues to the underlying physical mechanism (e.g., aquifer compaction vs. thermal stress) [Far+24].

Building on the highlighted background in PS-InSAR data clustering, the article describing the work published on the journal "Computers & Geoscience" is reported in Chapter 4.

2.2.3 Dynamic Graph Discovery: Networks in Motion

While InSAR deals with physical space, Spatio-Temporal discovery also extends to topological space: the complex networks of social interactions, legal citations, and criminal organizations. In these "dynamic graphs", the "location" of an entity is defined by its connections, which evolve continuously over time. **Temporal Graph Neural Networks** (TGNNs) represent the frontier of learning on dynamic graphs [Lac+, ZYW24]. Unlike static Graph Neural Networks (GNNs) that assume a fixed adjacency matrix, TGNNs must process a stream of node additions, edge modifications, and attribute changes. The field distinguishes between **Discrete-Time Dynamic Graphs** (DTDGs), which view history as a sequence of snapshots (e.g., annual court citation graphs), and **Continuous-Time Dynamic Graphs** (CTDGs), which model events as they happen (e.g., instant financial transactions). The latter is computationally demanding because every event potentially triggers a message-passing operation across the network. To address this scaling challenge, frameworks like **Zebra** have been introduced. Zebra optimizes the "temporal message passing" scheme by establishing a theoretical connection to **Temporal Random Walks**. It utilizes a **Temporal Personalized PageRank** (T-PPR) metric to identify and select only the most influential temporal neighbors for each node. This selective aggregation allows Zebra to ignore irrelevant historical noise, achieving speedups of up to two orders of magnitude compared to standard TGNNs while maintaining or improving predictive accuracy [Li+23b]. The legal system is a prime example of a complex adaptive system where the "meaning" of a node (a legal case) is determined by its temporal relationships (citations) with other nodes [Cou+21].

Research into legal citation networks has moved beyond simple centrality measures. A recent methodological approach in **Legal Judgment Prediction** (LJP) involves the introduction of explicit "time nodes" into the graph structure. In this architecture, every legal case is connected not only to the cases it cites but also to a node representing the year of its judgment. This allows the GNN to learn temporal embeddings, effectively capturing the "Zeitgeist" or the evolving judicial standard of a specific era. For instance, the severity of sentencing for certain crimes may drift over decades; the time node allows the model to adjust its prediction based on the temporal context. When combined with pre-trained language models like **XLNet** [Yan+20b],

these time-aware graph models have achieved Macro F1 scores of 75% in predicting case outcomes, outperforming text-only baselines [Kha+23].

A persistent challenge in LJP is distinguishing between "confusing law articles", such as statutes that are textually similar but legally distinct (e.g., different degrees of theft). To solve this, researchers utilize **Graph Attention Networks** (GATs). The model builds a graph where nodes are law articles and edges represent their co-occurrence or textual similarity. The GAT then learns to attend to the subtle differences in the "neighborhood" of each statute, effectively disentangling them in the embedding space. This "distinction extractor" is fused with the case-fact graph to provide precise charge predictions [Zha+22].

Criminal networks (drug trafficking syndicates, gangs) operate covertly, making their structures difficult to map. However, ST analysis of seized communication logs reveals striking patterns in their dynamics, particularly during periods of external stress (e.g., police investigations or turf wars). A seminal study on drug trafficking networks monitored network metrics over time and found that during the "stage of highest activity" (often preceding a major shipment or reacting to a threat), the network structure undergoes a phase transition [AA24]:

- In-Degree Entropy Minimizes: The sources of incoming communication become highly concentrated (everyone reports to the boss).
- Out-Degree Entropy Maximizes: The targets of outgoing communication become highly diverse (the boss issues orders to many subordinates).

This divergence indicates a centralization of command but a decentralization of execution. The study noted that metrics like *betweenness centrality* could change by up to 90% during these active phases. This insight is operationally critical: law enforcement can monitor these entropy shifts in surveillance data to predict imminent criminal events or identify key leaders who emerge only during crises [AA24].

2.2.4 The Frontier: Spatio-Temporal Foundation Models

The most ambitious development in the 2024 – 2025 period is the quest for **Spatio-Temporal Foundation Models** (STFMs). Inspired by the success of Large Language Models (LLMs) like GPT-4, researchers are attempting to build general-purpose models that can "understand" the physics and dynamics of the world across diverse domains – from urban traffic and climate systems to human mobility and epidemiology [Goo+25, Lia+25].

The primary barrier to a "GPT for Spatio-Temporal Data" is heterogeneity. Text is composed of discrete tokens (words) with a fixed vocabulary. ST data, however, is continuous, multi-modal (raster, vector, graph), and scale-dependent. A model trained on high-frequency traffic data (seconds) struggles to transfer knowledge to low-frequency climate data (months). To solve this, the **Factorized Spatio-Temporal Learning** (FactoST) architecture proposes a decoupling strategy [Zho+25b]. It posits that temporal patterns (e.g., seasonality, trends, periodicity) are somewhat universal, while spatial correlations are domain-specific.

- Stage 1 (Universal Temporal Pretraining): A lightweight backbone is pre-trained on massive, diverse datasets to learn general temporal dynamics. It ignores spatial topology.
- Stage 2 (Spatial Adaptation): A domain-specific adapter is attached to inject spatial knowledge (e.g., the road network graph or the sensor grid topology).

This "factorized" approach drastically reduces the parameter count (by 46%) and allows the model to adapt to new domains with minimal data (Few-Shot Learning), achieving state-of-the-art forecasting accuracy [Zho+25b].

Another architecture, **UniFlow**, tackles the divide between grid-based data (e.g., crowd density heatmaps) and graph-based data (e.g., traffic flow on roads) [Yua+25]. It introduces a multi-view spatio-temporal patching mechanism. This mechanism breaks disparate data types into standardized "patches" (sequences of ST tokens), allowing a single Transformer-based model to process them uniformly. By creating a shared embedding space for grids and graphs, UniFlow enables cross-pollination of knowledge: patterns learned from crowd flows can improve traffic predictions, and vice-versa [Yua+25].

Given that much ST data is sensitive (e.g., individual mobility traces, medical records), centralized training of Foundation Models is often legally impossible; **Federated Learning** has emerged as a key enabler. Frameworks like FFTS (**Federated Foundation Time Series**) allow organizations to train local models on private data while sharing only model updates (gradients) with a central server. Novel regularization mechanisms align the disparate feature spaces of these local models, allowing the global Foundation Model to learn from heterogeneous, decentralized data without ever "seeing" the raw private information [Che+25].

2.2.5 Grand Challenges and Open Problems

Despite the remarkable progress, significant hurdles remain in scaling ST discovery to the exascale level. The "All-Pairs" bottleneck remains a fundamental limit [LDR20, Lia+25]. Algorithms like Hierarchical Clustering or the Attention mechanism in Transformers scale quadratically ($O(N^2)$) with the number of data points. For a global climate model or a city-wide trajectory dataset with billions of points, this is intractable. Approximate methods (like the T-PPR in Zebra or Sparse Matrix Attention) are alleviating this, but a true $O(N)$ solution for high-fidelity ST clustering remains elusive [Li+23b].

Integrating heterogeneous sources faces a "semantic gap." A "node" in a social graph and a "pixel" in a satellite image are fundamentally different entities. Merging them requires not just technical alignment (resampling) but ontological alignment. Current Foundation Models often rely on "lossy" preprocessing to force data into a common format. The development of native multi-modal architectures that can respect the intrinsic geometry of each data type while fusing them at a semantic level is a critical research direction [Goo+25].

Most current methods are correlational. They can predict where a crime will happen or when a landslide will accelerate, but they rarely explain why. Moving from predictive correlation to causal inference — understanding the drivers of the system — is the next frontier. This is particularly vital in policy-making applications (e.g., "Will building this road cause more traffic elsewhere?"), where "black box" predictions are insufficient.

Attempting to tacke the explainability issue in the data clustering field, the article published on the journal "Computers & Geoscience" on the clustering of PS-InSAR TS data from satellite images explains how the problem was approached in Chapter 4.

To summarize the literature for Spatio-Temporal Pattern Discovery, Table 2.3 reports the key innovation for each domain, its main mechanism, and the primary benefits that came together with the novel method.

Domain	Key Innovation	Mechanism	Primary Benefit
Clustering	ST-DBSCAN	Dual-ε (Space/Time)	Detection of moving clusters; noise rejection.
	STkM	Unified Objective Function	Tracking dynamic groups with static identity
	Hierarchical	Adaptive Scaling (S_D, S_T)	Multi-scale analysis; cyclical time handling
InSAR	VAE-Clustering	Deep Generative BSS	70-84% noise reduction; blind separation
	UMAP-HDBSCAN	Dim-Reduction + Density	Visualization of massive stacks; breakpoint detection
Graphs	Zebra (TGNN)	Temporal PageRank (T-PPR)	100x speedup; efficient message passing
	Legal Time Nodes	Explicit Temporal Embedding	Capture of evolving legal standards/ precedents
Foundation Models	FactoST	Factorized Pretraining	Decoupling universal time from specific space
	UniFlow	Multi-View Patching	Unifying grid (raster) and graph (vector) flows

Table 2.3: Summary of Key Technologies & Trends (2015-2025).

2.3 Data Structures for Efficient Storage & Querying

The trajectory of data management between 2010 and 2025 has been defined by a fundamental conflict: the exponential growth in the volume of TS and sequence data versus the stagnant growth in memory bandwidth and random-access latency [LPK22]. This era, characterized by the ubiquity of the Internet of Things (IoT), high-frequency financial telemetry, and large-scale genomic sequencing, has necessitated a departure from traditional archival compression techniques toward **Compact Data Structures** and **Succinct Representations** [AKB23]. These structures are designed not merely to reduce storage footprints but to enable high-performance analytics directly on the compressed representation, thereby mitigating the "von Neumann bottleneck" where data movement costs exceed computation costs [MHM25].

Historically, the management of sequential data relied on a "compress-then-decompress" paradigm. Algorithms like Gzip [Deu96] or Snappy [Sna] were applied to large blocks of data (e.g., 64KB or 1MB pages) [Zen+23]. While effective for reducing disk usage, this approach

imposed a severe latency penalty for granular queries: retrieving a single sensor reading or finding the rank of a specific value required decompressing an entire block, saturating CPU caches and memory buses with irrelevant data [HPZ11]. The research landscape over the last decade and a half has systematically dismantled this model, replacing it with **fine-grained, random-access-capable structures that operate close to the information-theoretic lower bound of entropy**. This approach to data interrogation is crucial for the Big Data field, as more and more data needs to be stored in Data Lakes and Data Warehouses to be subsequently processed. Time Series data, especially fine-grained sampled multivariate data, can be expensive to store in raw formats like CSV, as the series can be non-repetitive and require high precision, thus making the data itself difficult to compress efficiently. That is why *reference to my project at Coruna*. Moreover, having a compressed query-able pool not only reduces the infrastructure necessities in terms of hardware and maintenance, but it also diminishes overall carbon emissions, favoring the environment and allowing industries to move towards fully green datacenters.

The latest advancements in the field are causing the distinction between a machine learning model and a data structure to blur, thanks to the emerging frontier of Homomorphic Compression and Learned Indexes, where the best way to represent data is learned **directly from the data itself** through the application of the fundamental principles of ML [Sah+, Tan+25].

2.3.1 Theoretical Foundations: Succinctness, Entropy, and the Rank/Select Primitive

To understand the advancements in Time Series storage, one must first ground the discussion in the theoretical definitions of succinctness that drove academic research during this period. A data structure is termed "succinct" if it requires $Z + o(Z)$ bits of space, where Z is the information-theoretic optimal number of bits needed to store the data (typically bounded by the zeroth-order empirical entropy H_0), and supports query operations in time complexity effectively independent of the dataset size (e.g., $O(1)$ or $O(\log|\Sigma|)$).

The foundation of succinct sequence representation rests on two bit-vector operations and three main operations: **Rank**, **Select**, and **Access**.

- Rank $q(x, i)$: Calculates the number of occurrences of symbol x in the prefix $S[1..i]$ of the sequence.
- Select $q(x, j)$: Identifies the position of the j-th occurrence of symbol x in the sequence.
- Access (i): Retrieves the symbol at position i.

Between 2010 and 2015, research solidified the bounds for these operations. While constant-time solutions for binary strings were well-known, the challenge lay in extending these to the large alphabets typical of numerical time series (e.g., the set of all unique temperature values in a year-long log). The theoretical objective was to achieve these operations in $O(\log|\Sigma|)$ time — where Σ is the alphabet size — without inflating the space requirement beyond the compressed size of the raw data [Bab+15, Shu20].

A critical insight developed in this period was the relationship between compression and combinatorial structure. Research demonstrated that structures like the **Wavelet Suffix Tree** could simultaneously capture the combinatorial properties of substrings (enabling efficient pattern matching) and the random-access capabilities of succinct indexes. This duality meant that a

single data structure could serve as both a compressor (reducing space to $O(n)$ or $O(n \log(\sigma))$ bits) and an index (supporting complex queries like identifying the k-th lexicographically smallest suffix of a substring) [Bab+15]. This convergence of compression and indexing is the defining characteristic of modern succinct data structures, distinguishing them from purely archival formats that are opaque to query engines until decompression.

2.3.2 The Wavelet Tree Ecosystem

The Wavelet Tree, originally introduced to generalize rank/select operations to non-binary alphabets, became a focal point of intense optimization between 2012 and 2025. The structure recursively decomposes a sequence based on value ranges [Bri+09]. For a sequence over the alphabet Σ, the root node represents the entire range of values. A bit vector at the root marks whether each element in the sequence belongs to the lower or upper half of the range. This process is repeated recursively for the left and right children until the leaves represent individual symbols [Bab+15].

By 2012, it became apparent that the standard pointer-based implementation of Wavelet Trees was ill-suited for modern CPU architectures. The recursive nature of the tree required pointer chasing — traversing nodes allocated non-contiguously in memory — which induced frequent cache misses and stalled the CPU pipeline. A seminal study from Aarhus University [KP] highlighted these hardware-based performance penalties, noting that the theoretical $O(\log|\Sigma|)$ complexity masked a high constant factor dominated by memory latency. This realization spurred the development of the **Wavelet Matrix**, a restructuring of the Wavelet Tree that eliminates pointers. In a Wavelet Matrix, the bit vectors for each level of the tree are concatenated into a single contiguous block per level. Navigation is performed via arithmetic calculations on rank values rather than pointer dereferencing. This layout significantly improved spatial locality, allowing the prefetcher to load subsequent-level data efficiently [CN12]. Experiments demonstrated that Wavelet Matrices achieved the same space efficiency as standard Wavelet Trees but with query times that were robust against large alphabet sizes, as they avoided the randomized memory access patterns of the pointer-based variants [CN12].

While standard Wavelet Trees handle general sequences, Time Series data often consists of integers with specific distributions. The **I**nteger **W**avelet **T**ree (IWT) was adapted to handle sequences of unique integers or permutations, effectively treating the integer domain as the alphabet [Sah+]. A major breakthrough in the 2020s was the realization that IWTs could exploit the "near-sortedness" inherent in many Time Series (e.g., timestamps, monotonically increasing counters). When a sequence is sorted or nearly sorted, the bit vectors in the upper levels of the Wavelet Tree become highly repetitive sequences of 0s followed by 1s. By applying Run-Length Encoding (RLE) or compressed bitmap techniques (like Roaring Bitmaps) to these node vectors, the space complexity of the IWT drops dramatically. Research presented at VLDB 2025 benchmarked the use of IWTs for "Physical-to-Sorted Mapping" in learned indexes [Sah+]. The study found that for highly sorted data, an IWT could occupy 84% less space than a B+-tree while still supporting the necessary rank and select operations to map a key's sorted rank to its physical location. This efficiency makes the IWT a formidable competitor to traditional indexes in memory-constrained environments, offering a viable middle ground between the lookup speed of a B+-tree and the compactness of a sorted array [Sah+]. To further mitigate the depth-related latency of binary Wavelet Trees, researchers explored T-way Wavelet Trees (or multi-ary Wavelet Trees) [Shu20]. Instead of a binary split (0 vs. 1), each node partitions the alphabet into T sub-ranges (e.g., 4 or 8). This reduces the tree height from $\log_2|\Sigma|$ to $\log_T|\Sigma|$, effectively

acting as a B-tree adaptation for succinct structures. While this increases the complexity of the node-level processing (requiring multi-symbol rank operations), the reduction in cache misses typically results in a net performance gain for large alphabets [Sah+, Shu20].

The practicality of succinct data structures is often limited by the time required for their construction. Constructing a Wavelet Tree sequentially takes $O(n \log|\Sigma|)$ time. For massive time series datasets, this latency is unacceptable. Parallel algorithms developed during this period demonstrated that Wavelet Trees could be constructed in $O(n)$ work and polylogarithmic depth [Shu20]. By partitioning the input sequence and constructing sub-trees in parallel before merging them, these algorithms allow succinct structures to be rebuilt dynamically in high-throughput database environments, removing the "static data only" limitation that had previously constrained their adoption [Shu20].

2.3.3 The Gorilla Lineage: Streaming Compression for Floating-Point Data

While Wavelet Trees address the need for indexing and random access, the primary challenge for Operational Historians and Time Series Databases (TSDBs) is write throughput. From 2015 onwards, the industry coalesced around a family of "XOR-based" streaming compression algorithms, originating from Facebook's Gorilla database [Pel+15]. These algorithms prioritize ingestion speed and sequential compression ratio over random access. The release of the Gorilla paper in 2015 marked a watershed moment. The algorithm was designed to compress 64-bit double-precision floating-point values, which are notoriously difficult to compress using generic text algorithms (like LZ77 [LZ77]) due to the high entropy of the mantissa bits [AKB23]. Gorilla leveraged a domain-specific observation: in physical TS (server metrics, sensors), value v_t is highly correlated with value $v_{\{t-1\}}$. Mechanism:

- XOR Difference: The algorithm computes $\Delta = v_t \oplus v_{\{t-1\}}$.
- Zero-Encoding: If Δ is zero (values are identical), a single 0 bit is written.
- Control Bits: If Δ is non-zero, the algorithm calculates the number of leading zeros (LZ) and trailing zeros (TZ).
- Bit Packing: If the LZ and TZ match the previous value's pattern, only the meaningful XORed bits are written. Otherwise, new control bits (5 bits for LZ, 6 bits for length) are written to define the new meaningful block.

This mechanism exploits the fact that even if a float changes slightly (e.g., 12.01 to 12.02), the high-order bits (exponent and sign) and many mantissa bits remain identical, resulting in an XOR pattern dominated by zeros. Gorilla achieves compression ratios of 10x-15x on typical IT metric data and became the standard for systems like Prometheus, InfluxDB, and TimescaleDB [Pro26, Tim26, Inf, MBP25].

In 2022, the Chimp algorithm [LPK22] identified a sub-optimality in Gorilla: the assumption that the "previous value" is always the best predictor. Chimp observed that leading zero counts (LZ) follow a recognizable distribution and that the previous value's XOR pattern often fails to capture periodic regularities [Li+23c]. The Chimp Innovations:

- Prediction Buffer: Chimp employs a small circular buffer (of size 128 in Chimp128) to store recent values. Instead of strictly XORing with $v_{\{t-1\}}$, it searches the buffer for a candidate that yields a smaller XOR difference (more leading zeros).
- LZ Prediction: Chimp avoids explicitly storing the Leading Zero count for every value. It uses the previous value's LZ count as a prediction; if the current LZ matches the prediction, no bits are spent on metadata.

Benchmarks demonstrated that Chimp reduces the storage footprint of Gorilla by an additional 50% for many datasets while maintaining or even improving decompression speed [LPK22]. Crucially, Chimp supports decompression speeds of over 3 GB/s, enabling "decompression-on-the-fly" during query execution. This capability allows TSDBs to keep data compressed in the CPU cache (L3) and decompress it only in registers, effectively expanding the effective cache size [BMG18, Li+23c].

The lineage continued with Patas [Lia+24] (integrated into DuckDB) and Elf [Li+23d]. Patas optimizes for modern superscalar CPUs by removing bit-level dependencies that stall pipelines. It avoids the variable-length bit packing of Gorilla in favor of byte-aligned operations where possible, trading a small amount of space for significantly faster decoding. Elf focuses on preserving high precision for scientific data while allowing for approximate queries, further blurring the line between lossless and lossy representation.

2.3.4 Integer Time Series and IoT: The Sprintz Approach

While Gorilla dominated the server/cloud metric space (floats), the IoT sector required efficient compression for low-power devices generating integer data (e.g., ADCs, counters). In 2018, the Sprintz algorithm was introduced [BMG18] to address the constraints of embedded systems: limited memory ($<$ 1KB), low compute power, and the need for energy efficiency. The algorithm improved on various fields of the literature:

- Forecasting and Residuals: Sprintz differs from Gorilla by focusing on forecasting rather than XOR.
- Predictive Models: It employs lightweight predictors such as Delta ($v_t - v_{t-1}$) or Delta-of-Delta ($(v_t - v_{t-1}) - (v_{t-1} - v_{t-2})$). The choice of predictor is dynamic or fixed based on the variable type.
- ZigZag Encoding: The residuals (errors) are ZigZag encoded [BMG18] to map negative numbers to positive integers, ensuring that small errors result in small integers with many leading zeros.
- SIMD (Single Instruction, Multiple Data) instructions for bit packing: one of the most innovative changes introduced by Sprintz is the usage of vectorized operators (e.g., AVX2, NEON) to speed up a range of operations:
  - Block-Based Encoding: Sprintz groups residuals into blocks (e.g., 8 or 16 values). It calculates the maximum number of bits (b) required to represent the largest error in the block.

– Parallel Packing: It then packs all values in the block using b bits. Unlike variable-bit-length encoding (like VarInt), which requires branching for every byte, the block-based approach allows SIMD instructions to pack/unpack multiple integers in a single CPU cycle.

This design enables Sprintz to achieve state-of-the-art compression ratios for multivariate integer time series while decompressing at rates exceeding 4 GB/s [BMG18]. Furthermore, the block headers provide a mechanism for coarse-grained random access: a query can skip over blocks of time without bit-level decoding, addressing a major weakness of pure delta chains [Gue+25].

2.3.5 Grammar-Based Compression: Mining and Anomaly Detection

A distinct thread of research views TS not as numerical sequences but as strings of symbols, leveraging Grammar-Based Compression to uncover structural patterns. This approach is less about minimizing storage bytes and more about enabling Pattern Mining and Anomaly Detection directly from the compressed structure [Sen+18]

Since grammar algorithms operate on discrete alphabets, continuous TS must be tokenized. **S**ymbolic **A**ggregate appro**X**imation (SAX) [Lin+07] is the standard preprocessing step. SAX normalizes the data (z-score), divides it into fixed-width windows with a technique called **P**iecewise **A**ggregate **A**pproximation (PAA), and maps the mean of each window to a symbol (e.g., 'a', 'b', 'c') based on Gaussian breakpoints [GLR17]. PAA is a dimensionality reduction technique where a TS of length n is divided into w equal-sized frames, and the mean value of the data within each frame is calculated [Lin+07]; those mean values are those being mapped to symbols. A critical refinement in this pipeline, which is also applicable to many grammar-based compression algorithms, is **numerosity reduction**. If a TS stays constant or changes slowly, SAX might generate a string like "aaaaabbbbb". Numerosity reduction collapses this to "a b" (recording offsets), preventing the grammar engine from wasting rules on trivial runs.

Another clever algorithm is Sequitur [HD25], a linear-time algorithm that constructs a **Context-Free Grammar** (CFG) from a string. A CFG is a formal grammar where every production rule is of the form $V \to w$, where V is a single non-terminal symbol [GG10]. In the context of Sequitur, the CFG is used to represent a long sequence as a set of hierarchical rules, "compressing" the sequence into its underlying structural logic [Sen+18]. To do this, Sequitur enforces two constraints: (i) Digram Uniqueness (no pair of adjacent symbols appears more than once in the grammar) and (ii) Rule Utility (every rule is used at least twice) [Sen+18].

GrammarViz 3.0. developed through the mid-2010s, applies Sequitur to SAX-encoded TS [Sen+18]. The resulting hierarchical grammar effectively captures the "normal" recurring patterns of the data (motifs) as high-level rules [Sen+]. Rule density is defined as the number of rules that cover a specific point. In the field of Anomaly Detection, the theoretical underpinning is Kolmogorov Complexity [KN21]. Anomalies are problematic from a compression point of view since a random (anomalous) sequence is **incompressible**. In GrammarViz, anomalies manifest as substrings that cannot be efficiently covered by the induced grammar rules. By calculating the rule density, the system can identify discord/anomalies in linear time. This "compression-as-learning" paradigm allows for zero-shot anomaly detection without training sets [Sen+].

Re-Pair (Recursive Pairing) is another grammar induction algorithm that often yields better compression ratios than Sequitur by globally replacing the most frequent pair in each iteration [BJP17]. However, Re-Pair's memory consumption historically limited its use on massive datasets. Recent work in 2024 addressed this scalability bottleneck with Recursive Re-Pair (Re2Pair) [Kim+24]. Re2Pair (and related parallel implementations) utilizes **Prefix-Free Parsing** (PFP) to preprocess the text, allowing the grammar construction to be parallelized across multiple threads. This reduces peak memory usage by over 40% and speeds up construction by up to 79%, making grammar-based compression viable for terabyte-scale genomic or sensor datasets [Kim+24].

In many industrial and scientific applications, the challenge is not compressing a single unique time series, but compressing a collection of highly similar sequences (e.g., 1,000 sensors of the same model, or 10.000 human genomes). Standard compressors (e.g., Gunzip) process each file independently, missing the massive inter-sequence redundancy. **R**elative **L**empel-**Z**iv (RLZ) was developed to exploit this similarity [KPZ10, PZ20]. RLZ compresses a target sequence T relative to a Reference Sequence R. Instead of a sliding window (as in LZ77 [LZ77]), the dictionary is the static reference R. T is parsed into a sequence of factors (pos, len) representing substrings found in R. A crucial advantage of RLZ is its support for random access. To retrieve the value at $T[i]$, one locates the factor covering position i and retrieves the corresponding data from the reference R. Since R is kept in memory, this is an $O(1)$ operation (plus the time to search the factor index), contrasting sharply with the linear scan required for LZ77 [HPZ11]. For complex collections where a single reference is insufficient (e.g., a mix of data from different sensor types), **Hierarchical RLZ** (HRLZ) constructs a tree of references [Bil+22]. The root is a "global" reference; children are compressed relative to the root, and leaf nodes (actual data) are compressed relative to their parents. This hierarchy allows the system to capture both global trends and cluster-specific variations. Benchmarks on bacterial genomes showed HRLZ achieving **double the compression ratio of standard RLZ** with negligible impact on decompression speed [Bil+22].

The efficacy of RLZ hinges on the reference. For TS, simply picking one sequence is often suboptimal. Research has focused on **Synthetic Reference Construction**: generating an artificial sequence that concatenates the most frequent patterns (motifs) found across the entire collection. This aligns with the "Dictionary Learning" approach in signal processing, ensuring that the reference provides high coverage for any sequence in the dataset [KPZ].

Following a similar approach, Chapter 5 illustrates the work carried out at the University of A Coruña, in Spain, to attempt improving on RLZ's default reference creation through "educated guesses" while respecting the computational efficiency of the original algorithm through the usage of Suffix Arrays.

2.3.6 Physical Storage and Columnar Layouts

The transition from algorithms to deployed systems is best exemplified by the rise of **Columnar Storage Formats**: Apache Parquet, ORC, and Avro have become the de facto standards for data lakes, embedding many of the compact data structure principles discussed above directly into their file layouts [NS25]. Benchmarks from 2024 – 2025 reveal distinct performance characteristics [Zen+23], summarized in Table 2.4:

Feature	Apache Parquet	Apache ORC	Apache Avro
Storage Model	Columnar	Columnar	Row-based
Compression	High (Snappy, Gzip, Zstd)	Highest (Lightweight indexes)	Moderate
Encoding Support	Dict, RLE, Bit-Packing	Dict, RLE	ZigZag, VarInt
Write Speed	Slow (Heavy encoding overhead)	Moderate	Fastest (Append-friendly)
Read/Query	Fastest (Projection pushdown)	Very Fast	Slow (Full scan required)
Use Case	OLAP, Analytics, Spark	Hadoop/Hive Ecosystem	Streaming (Kafka), ETL

Table 2.4: Format Architectures and Trade-offs by feature.

A key feature enabling "compressed domain" performance in Parquet and ORC is **Predicate Pushdown** via metadata statistics. This feature consists of a first step where data is divided into Row Groups (e.g., 128MB). The file footer contains the Min, Max, and Null Count for every column in every Row Group. When a query engine (Spark, Presto, DuckDB) executes a filter (e.g., timestamp $> T$), it inspects the footer first. If a Row Group's Max_Timestamp $< T$, the entire group is skipped. This avoids disk I/O and decompression entirely for irrelevant data blocks [Zen+23]. ORC takes this further with "Zone Maps" (indices every 10.000 rows), allowing for even finer-grained pruning than Parquet's Row Groups [Zen+23].

For columns with low cardinality (e.g., "Device_ID"), Parquet automatically employs **Dictionary Encoding**, which consists of replacing the values with small integer keys. Moreover, modern query engines process these keys using vectorized instructions: a filter Device_ID $=$ 'Sensor_A' is translated into an integer comparison Key $= 5$; then, the engine scans the packed integer keys using SIMD operators, achieving massive throughput. This effectively implements a "succinct" operation — operating on the compressed representation — within a standard file format [Liu+24a, Zen+23].

2.3.7 Direct Query Processing: The Frontier of Homomorphic Compression

The ultimate goal of succinct representations is to support not just search (Rank/Select) but computation (Aggregation, Join) on compressed data. This field, often termed **Compressed Domain Computing**, saw significant theoretical and practical steps between 2020 and 2025 [Zen+23]. **CompressIoTDB** [Tan+25] introduced a formal framework for Homomorphic Compression in Time Series in 2025. The core idea is to define compression schemes as algebraic homomorphisms, defined as follows:

Definition 2.1 (Homomorphic Compression Functions)

A compression function C is homomorphic for an operator $\oplus$ if the result of the operation on compressed data can be mapped to the result on uncompressed data:

$$C(x \oplus y) \approx C(x) \diamondsuit C(y) \tag{2.3}$$

For example, if data is delta-encoded, the sum of a range can be computed by summing the deltas and applying a correction factor, rather than fully reconstructing the absolute sequence [Tan+25]. By deferring decompression until the final result is needed (Late Decompression), CompressIoTDB reduces memory usage by 20% and improves query throughput by over 50%. The system introduces a "CompColumn" structure in memory that supports window-based aggregation directly on the encoded bits [Tan+25].

The **NeaTS** algorithm, published in 2024, represents a hybrid approach between lossy compression and succinct indexing [Gue+25]. It approximates Time Series using a sequence of nonlinear functions (polynomials, exponentials) rather than simple deltas. Unlike delta-chains (Gorilla), which require sequential traversal, NeaTS supports $O(1)$ random access. To retrieve value v_t, one identifies the function f_k covering time t, calculates $f_{k(t)}$, and adds the stored residual. NeaTS also offers a tunable trade-off: by discarding residuals, it becomes a highly compact lossy representation; by keeping quantized residuals, it becomes lossless. This flexibility allows it to serve as both an archival format and a real-time operational store [Gue+25].

The **Succinct** data store leverages **Compressed Suffix Arrays** (CSA) to enable regex and substring search on unstructured logs and semi-structured events. Unlike Parquet, which optimizes for columnar scans, Succinct optimizes for search. It enables "search on compressed data" by navigating the suffix array (via the Burrows-Wheeler Transform) without inflating the text. However, benchmarks indicate that while Succinct excels at counting and locating patterns, it can be slower than Parquet for aggregation queries that require accessing every value in a column, revealing a "No Free Lunch" trade-off between search-optimized and scan-optimized succinct structures [Wan+24a].

2.3.8 Emerging Frontiers: Learned Indexes and Neural Compression

As we look toward the latter half of the 2020s, the boundary between "Data Structure" and "Machine Learning Model" is dissolving [Sah+]. The concept of **Learned Indexes**, introduced around 2018 and matured by 2025, proposes replacing structures like B-Trees and Wavelet Trees with **neural networks**. Instead of traversing a tree to find the position of a key, a model predicts the position: pos = Model(key). Learned Indexes require a mapping from the predicted rank to the physical memory address. **Compressed Wavelet Trees** (IWTs) are currently the most efficient way to store this mapping, creating a symbiotic relationship where ML models provide the "search" logic and succinct data structures provide the "storage" logic [Sah+]. DL models (Autoencoders, RNNs) are also being investigated for TS compression. While they achieve extremely high compression ratios by "learning" the generative process of the data, they currently suffer from slow inference (decompression) speeds compared to algorithmic approaches like Sprintz or Chimp [LPK22]. However, hardware accelerators (TPUs/NPUs) may soon make neural decompression viable for real-time systems.

2.4 Data Fusion and Semantic Matching

The discipline of data management is currently navigating its most significant transformation since the relational revolution of the 1970s. For decades, the field was governed by deterministic logic, rigid schemas, and symbolic manipulation. The primary objective was structural alignment – forcing disparate data sources into a pre-defined "golden schema" through **extract-transform-**

load (ETL) pipelines and rule-based matching. However, the first half of the 2020s has witnessed a fundamental paradigm shift toward probabilistic, semantic, and generative approaches. This transition is not merely a change in tooling; it represents a cognitive shift in how machines perceive, interpret, and unify information.

The central challenge of this era is **Semantic Data Fusion**: the automated integration of heterogeneous data into a unified, consistent truth. Unlike its predecessors, which focused on syntactic consistency (e.g., ensuring date formats align), semantic fusion grapples with the inherent ambiguity of meaning. It asks whether "Apple Inc." in a financial database refers to the same real-world entity as "Apple" in a fruit wholesaler's inventory, or if the "Jaguar" in a biodiversity report corresponds to the "Jaguar" in a luxury car dealership's sales log [Mos+25].

Historically, this problem — known variously as **Entity Resolution** (ER), **Record Linkage**, or **Deduplication** — was approached as an engineering task. Systems like Magellan [Kon+16] epitomized this era, providing robust frameworks for feature engineering where human domain experts crafted rules and similarity metrics (e.g., Jaccard similarity > 0.8) to train supervised classifiers. While effective for structured data within narrow domains, these symbolic systems hit a "glass ceiling" when confronted with the messy, unstructured reality of the modern web and enterprise data lakes. They lacked the semantic depth to understand context, struggled to generalize across domains, and required prohibitive amounts of manual maintenance [Kon+16].

The inflection point arrived with the maturity of DL and the advent of the Transformer architecture. Between 2020 and 2025, the research landscape exploded with innovations that moved from static word embeddings to context-aware language models like **BERT** and **RoBERTa** [Dev+19, Liu+19, ZS24], and finally to Generative Large Language Models (LLMs) like **GPT-4** and **Llama** [Ope+24, Tou+23]. These models brought with them the ability to perform "zero-shot" reasoning: matching entities they had never seen before by leveraging vast repositories of pre-trained world knowledge, an emerging capability the model would develop without fine-tuning.

Simultaneously, the scope of Data Fusion expanded beyond static text. In high-stakes domains (e.g., legal analytics and social computing), the definition of a "record" evolved to include multimodal signals such as audio from courtroom proceedings, video evidence, and temporal knowledge graphs that track the evolution of case law over time. This necessitated the development of Multimodal Semantic Fusion, where vector spaces of pixels, waveforms, and tokens must be aligned to predict outcomes or detect deception [Bal+25].

Furthermore, as these systems became more powerful, they also became more opaque. The "black box" nature of deep neural networks created a crisis of trust, birthing the field of **Explainable Entity Matching** (XAI). Stakeholders in regulated industries could no longer accept a "match" decision without a rationale. This drove research into feature attribution methods like **SHAP** [NSK25] and **LIME** [RSG16], and eventually, to the use of LLMs themselves as "explainers" that verbalize the reasoning behind a decision [Wad+24].

Finally, the generative capabilities of LLMs introduced a new architecture for data quality: **Retrieval-Augmented Generation** (RAG) [Lew+21]. RAG moved beyond simple query answering to active data cleaning and imputation, using the "parametric memory" of LLMs combined with the "non-parametric memory" of enterprise data lakes to fill missing values and correct errors with unprecedented accuracy [Nae+24].

2.4.1 The Evolution of Entity Matching: From Symbolic Engineering to Generative Reasoning

Entity **M**atching (EM) is the cornerstone of **Data Integration**. It is the process of identifying records that refer to the same real-world entity across different data sources. The evolution of EM from 2020 to 2025 can be categorized into three distinct generations: (i) the Symbolic Generation (Machine Learning), (ii) the Neural Generation (Pretrained Language Models, PLMs), and (iii) the Generative Generation (LLMs).

Before the dominance of DL, EM was primarily a feature engineering problem. The state-of-the-art was represented by systems like Magellan, developed at the University of Wisconsin-Madison [Kon+16]. Magellan was not merely an algorithm but an end-to-end management system designed to guide users through the EM pipeline: **blocking** (reducing the search space), **matching**, and **debugging**. The system operated on a clear, structured workflow:

1. Attribute Selection: The user identified relevant columns (e.g., Title, Price, Brand).
2. Similarity Computation: The system applied library-based similarity functions (e.g., Levenshtein distance for names, absolute difference for prices) to generate a feature vector for each pair of records.
3. Classification: Standard supervised learning algorithms — Random Forests (RF), Support Vector Machines (SVM), or Decision Trees — were trained on these vectors to classify pairs as "match" or "non-match".

While Magellan provided a rigorous framework, it suffered from the **Semantic Gap** [Mos+25]. A string distance metric calculates the orthographic overlap between "iPhone 12" and "Apple Smartphone," but it captures no semantic relationship. To a symbolic system, these strings are distant; to a human, they are synonymous. Research from the University of Mannheim and others [PDB23] demonstrated that while SVMs and Random Forests were computationally efficient, they plateaued in performance on textual data. For example, on the WDC Products benchmark [PDB23], symbolic baselines like Magellan consistently underperformed neural models, particularly on datasets with high textual heterogeneity. The reliance on manual feature definition also made these systems brittle; a schema change or a new data source often required a complete re-engineering of the pipeline.

The second generation of EM was defined by the introduction of DL and, specifically, the Transformer architecture. This era shifted the focus from manually crafting features to learning representations (embeddings) from raw data. The transition began with DeepMatcher in 2018, which utilized Recurrent Neural Networks (RNNs) to summarize attribute values into dense vectors [Xie+23]. DeepMatcher compared these vectors using element-wise similarity and concatenation, feeding the result into a classifier. This approach successfully captured some semantic nuance but was limited by the sequential nature of RNNs, which struggled with long-range dependencies and parallelization. The true breakthrough occurred with the application of **Pre-trained Language Models** (PLMs) like **BERT** [Dev+19] and **RoBERTa** [Liu+19]. The defining system of this period (2020 – 2022) is **Ditto** (**Deep Entity Matching with Pre-Trained Language Models**) [Li+20a]. Ditto fundamentally reformulated the EM task: instead of comparing attribute A from Record 1 with attribute A from Record 2, Ditto serialized the entire record into a single text sequence using a specific format exploiting separator tokens, e.g., the following string: `"[COL] Title [VAL] Samsung Galaxy [COL] Price [VAL] \$500..."`; the

combined records were then fed into a Transformer, which performed binary classification on the token embedding [Li+20a]. Furthermore, Ditto exploited other innovations to increase its performance:

- Domain Knowledge Injection: Ditto recognized that generic pre-training was insufficient for specialized domains. It allowed users to inject domain knowledge by tagging specific spans (e.g., "Product ID") with special tokens, forcing the self-attention mechanism to prioritize these critical segments [Li+20a].
- Summarization: BERT-based models typically have a token limit (e.g., 512 tokens). Ditto integrated a summarization module using TF-IDF or text-rank to retain high-information words while discarding noise, ensuring that long product descriptions did not truncate essential specifications [Li+20a].
- Data Augmentation: To improve robustness, Ditto adapted **Mixup Data Augmentation** (MixDA) techniques for text, applying operators like *span deletion*, *shuffling*, and *synonym replacement* to create "hard" negative examples during training. This forced the model to learn more discriminative features [Li+20a].

On standard benchmarks like the WDC Products dataset, Ditto and its derivatives (e.g., R-SupCon [PB22], which adds supervised contrastive learning) achieved state-of-the-art results. In scenarios with large training sets, these models achieved F1 scores exceeding 90%, outperforming symbolic baselines [Li+20a]. However, distinct limitations emerged. Research analyzing BERT's internal behavior in EM tasks revealed a "Black Box" issue. While the model successfully identified the structure of entity records, it did not rely heavily on the pairwise semantic similarity of tokens, but rather on complex, learned patterns in the final layers [Pag+22, Par+25]. More critically, these models exhibited poor Out-of-Distribution (OOD) Generalization. A Ditto model trained on "Electronics" data would fail catastrophically when applied to "Clothing" data, or even to "Electronics" data with unseen brands. This "unseen entity" problem became a focal point of research in 2023 – 2025 [Mos+25, Pag+22].

The third and current generation (2023 – 2025) leverages Large Language Models (LLMs) like GPT-4 [Ope+24], Llama 3 [Tou+23], and Mixtral [Jia+24]. This shift moves EM from a "classification" task to a "reasoning" task. Unlike PLMs, which require fine-tuning on thousands of labeled pairs, LLMs can perform EM with little to no task-specific training; this is known as **In-Context Learning**. By providing a prompt that includes the task definition and perhaps a few examples (shots), the LLM utilizes its pre-trained world knowledge to determine if two records match. The World Knowledge gives a remarkable advantage: an LLM knows that "NY" and "New York" are the same, or that "1TB" is equal to "1000GB," without needing explicit training examples. This enables LLMs to handle unseen entities and corner cases [PSB24]. Empirical studies on the WDC Products benchmark show that GPT-4, in a zero-shot setting, can achieve F1 scores competitive with fine-tuned PLMs on bibliographic data and outperform them on highly heterogeneous e-commerce datasets [PSB24].

The use of LLMs introduces a new variable: cost. Fine-tuning a small model (like **DistilBERT** [San+20]) is computationally cheap at inference time but expensive in terms of data labeling. Querying GPT-4 has no cost relative to data labeling; however, it presents a high inference cost. Recent extensive studies in 2025 comparing 8 matchers across 11 datasets quantified this trade-off [Zha+25]. They found that fine-tuned small models (like Ditto or **Unicorn** [Tu+23]) can perform on par with prompted large models (like GPT-4). Specifically, a fine-tuned 124M parameter model (GPT-2 based) achieved an F1 of 81.5, comparable to GPT-4o-Mini's 83.9, but

at orders of magnitude lower deployment cost [Zha+25]. This has led to a strategic bifurcation: (i) *High-Value, Low-Volume*: Use LLMs (GPT-4) for difficult, ambiguous cases where accuracy is paramount, and volume is low; (ii) *Low-Value, High-Volume*: Use fine-tuned PLMs (Ditto/ RoBERTa) for bulk processing of known entity types.

To bridge the gap, researchers have developed distillation techniques: instead of using the LLM for inference, they use the LLM to generate training data or explanations, which are then used to train a smaller student model [Wad+24]. Wadhwa et al. demonstrated that distilling the "reasoning traces" (natural language explanations of why a match occurred) from an LLM into a smaller entity matching model improved the student's OOD generalization by over 10% [Wad+24]. This effectively transfers the "reasoning" capability of the giant model into the efficient architecture of the small model.

Table 2.5 summarizes the model architectures developed for Entity Matching across the years, highlighting the evolution of the mainstream solutions in the field.

Feature	Symbolic (e.g., Magellan)	Neural / PLM (e.g., Ditto)	Generative / LLM (e.g., GPT-4)
Core Mechanism	Feature Engineering + Classifiers (RF/ SVM)	Sequence Classification + Attention (BERT)	Generative Reasoning + In-Context Learning
Input Representation	Structured Attribute Vectors	Serialized Token Sequences	Natural Language Prompts
Training Requirement	High (Feature selection + Labeling)	High (Fine-tuning on thousands of pairs)	Low (Zero-shot or Few-shot prompts)
Generalization	Poor (Fails on schema changes)	Moderate (Fails on unseen entities)	High (Leverages world knowledge)
Interpretability	Moderate (Feature importance weights)	Low (Black box attention maps)	High (Natural language explanations)
Inference Cost	Very Low	Low to Medium	High (Token-based costs)
Primary Failure Mode	Semantic Gap (Orthographic distance)	OOD Brittle (Unseen tokens)	Hallucination & Prompt Sensitivity

Table 2.5: Comparative Analysis of EM Architectures.

2.4.2 Explainability (XAI) in Semantic Data Fusion

As Entity Resolution systems migrated from transparent rules (e.g., "Match if Last Name matches AND DOB matches") to opaque neural networks with millions of parameters, the "Black Box" problem became a critical barrier to adoption. In regulated industries like banking (Anti-Money Laundering) and healthcare (Patient Matching), a "match" decision triggers significant consequences. Stakeholders require not just a prediction, but a justification.

While Transformer-based models like BERT produce state-of-the-art results, their decision-making process is notoriously difficult to interpret. Analysis of BERT's behavior in EM tasks reveals that simple attention mechanisms (which highlight which words the model is "looking at") are often misleading. The model's fine-tuning process modifies the last layers to recognize specific structural patterns of "matching" vs. "non-matching" records, rather than relying on a straightforward semantic similarity comparison of individual tokens. This means that a high attention weight on a token does not necessarily mean that that token was the cause of the decision [Pag+22]. To address this, researchers have applied model-agnostic XAI methods. The two dominant frameworks in the 2020 – 2025 period are **SHapley Additive exPlanations** (SHAP) and **Local Interpretable Model-agnostic Explanations** (LIME) [NSK25, RSG16].

SHAP is grounded in cooperative game theory. It treats the prediction as a "game" and the input features (tokens) as "players". The goal is to fairly distribute the "payout" (the prediction score) among the players based on their contribution. To do so, SHAP computes the marginal contribution of a feature across all possible coalitions of features. Doing so provides consistency and local accuracy: if a model relies 100% on the word "Inc." to classify a company, SHAP will consistently attribute the score to that token. On the other side, it is computationally expensive. Calculating exact Shapley values is NP-hard, requiring approximations (like KernelSHAP [NSK25]) that can still be slow for high-dimensional text data [RFS25].

LIME takes a different approach. It creates a "surrogate" model (usually a simple linear model) that approximates the behavior of the complex neural network locally around the specific data point being explained. To do so, it perturbs the input (e.g., removing words) and observes how the prediction changes, fitting a linear regression to these perturbed samples. It is significantly faster than SHAP for single-instance explanations; however, it lacks stability. Running LIME twice on the same instance can yield different explanations due to the randomness of the perturbation sampling. It also struggles with non-linear local boundaries [Sal+25].

Both SHAP and LIME suffer from a limitation in the EM context: they provide feature-level (token-level) explanations. Telling a user that the token "Pro" increased the match probability by a certain amount is useful, but it doesn't explain the concept. The lack of a human-readable explanation to simplify comprehension of the prediction drove the shift toward **Natural Language Explanations**. The most promising development in XAI for EM is the use of LLMs to "verbalize" the reasoning process. This approach treats explainability as a translation task: translating the model's internal state (or the input data) into a human-readable narrative [Wad+24] GPT-4 and similar open (or closed) access models can be prompted to generate structured explanations for their matching decisions, allowing everyone — even people that are not formed in I.T. subjects — to perform this task manually if needed [PSB24]. A practical example of this is the work carried out with Cyntia Valeria Biondi Russo for her Master's Thesis, where Qwen3 [Yan+25a] was exploited to perform various steps of a pipeline that started from cleaning raw data to performing an EM task by matching Curriculum Vitae (CVs) with Job postings. The CVs where then ranked by pertinance. The pipeline details are described in Section 6.3.

Beyond individual instances, LLMs are now used for aggregate error analysis. Instead of a human manually reviewing 500 false positives, an LLM can analyze the explanations of these errors and cluster them into high-level categories. For instance, through an LLM, one could receive insight into the weak points of their model, noticing patterns where the model fails to match properly. This automated diagnosis allows Data Engineers to rapidly identify weaknesses in their pipeline and adjust blocking rules or training data accordingly [PSB24].

Current research suggests that the future of XAI lies in Hybrid Frameworks. Pure LLM explanations can be "hallucinated" (the LLM might give a plausible-sounding reason that doesn't reflect its actual decision process [Li+25a]). Pure SHAP values are accurate but hard to interpret. The hybrid solution consists of the usage of SHAP to identify salient tokens (the evidence) and then pass these tokens to an LLM to generate a narrative description of why those tokens matter. This combines the mathematical fidelity of SHAP with the linguistic intelligibility of LLMs [Bab24, Sim24].

The idea behind this approach is similar to that behind **Retrieval-Augmented Generation** (RAG), which will be explored further in Section 2.4.4.

2.4.3 Multi-Modal and Temporal Analytics in Legal and Social Domains

While Entity Matching focuses on unifying identities, the broader field of Semantic Data Fusion increasingly deals with complex, multi-modal data streams. This is particularly evident in the legal and social analytics domains, where a "case" or an "event" is a nexus of text, audio, video, and time-series data.

Traditionally, **Legal Judgment Prediction** (LJP) was a text-processing task. Models analyzed court transcripts or case files to predict charges, relevant statutes, and sentencing outcomes. However, research in 2024 – 2025 has highlighted the inadequacy of this text-centric view. Real-world judgments often hinge on evidence that is not fully captured in text: the tone of a witness's voice, the facial micro-expressions of a defendant, or the visual details of a crime scene photo. A pioneering system in this space is **LegalEye**, a multimodal DL framework designed for deception detection in courtrooms [Bal+25]. LegalEye employs a **Late Fusion** architecture; it uses separate neural networks to process three distinct modalities:

1. Audio: Processes vocal features like pitch, tone, and Mel-frequency cepstral coefficients (MFCCs);
2. Visual: Analyzes video feeds using Convolutional Neural Networks (CNNs) to track Facial Action Units (FAUs) and emotion vectors;
3. Text: Analyzes transcripts using NLP models (like BERT) to detect linguistic markers of deception (e.g., "linguistic immediacy"). The outputs of these independent networks are then concatenated and passed through a fusion layer to make a final credibility assessment.

The system achieved remarkable detection rates: 97% for English, 85% for Spanish, and 86% for Tagalog. Crucially, the study revealed modality-specific dominance: visual features were the most influential predictors for English and Tagalog speakers, while audio features were more predictive for Spanish speakers. This underscores the necessity of multimodal fusion; a text-only model would have missed the primary signals of deception [Bal+25].

Beyond deception, multimodal LJP is expanding to include *Physical Evidence*. Cases involving torts or violent crimes often rely on non-written evidence (e.g., the condition of a weapon, the layout of a room). When this evidence is merely summarized in text, significant information loss occurs. Recent studies [Liu+25b] show that "Evidence-Based LJP" models that ingest raw multimodal data (images of evidence, audio recordings) significantly outperform "Fact-Based

LJP" models that rely solely on textual descriptions of that evidence. This finding is pushing the legal tech industry toward systems that can ingest the entire "digital dossier" of a case, not just the clerk's notes [Liu+25b].

Legal cases and social events are not static snapshots; they are evolving processes. A static Knowledge Graph (KG) might capture the fact `(Person A, sued, Person B)`, but it fails to capture the sequence: `(Person A, sued, Person B, t=1) -> (Person B, appealed, t=2) -> (Court, remanded, t=3)`. To address this, researchers have developed **Temporal Knowledge Graph Forecasting** [Lin+25]. The goal is to predict future links (events) based on the *history* of the graph.

A state-of-the-art model introduced in 2025 is **EvoReasoner** [Lin+25]. It addresses the limitation of LLMs, which struggle to reason over knowledge that evolves (e.g., asking "Who is the Prime Minister?" when the government just changed). EvoReasoner employs "global-local entity grounding" and "temporally grounded scoring". These two terms describe the process of decomposing multi-hop queries into temporal segments and validating each step against the state of the graph at that specific time. By combining temporal reasoning with KG evolution, EvoReasoner outperforms standard prompting-based baselines, effectively narrowing the gap between small, specialized models and massive, static LLMs on dynamic question answering tasks [Lin+25].

In the legal domain, this TKG approach is used to predict the trajectory of a case. By modeling the case as a temporal graph of events, systems can predict the likelihood of an appeal or a settlement [JC22]. The **Knowledge-Enhanced LJP** (K-LJP) framework treats judgment prediction as a multi-task learning problem [Li+25b]. It incorporates "label-level knowledge" (the legal definition of a charge) and "task-level knowledge" (the relationship between a charge and a penalty) into the temporal reasoning process. This enables the model to predict not only the final verdict but also the sequence of legal articles that will be cited [Li+25b].

2.4.4 Retrieval-Augmented Generation (RAG) and Data Quality

The integration of Generative AI into data pipelines has given rise to **Retrieval-Augmented Generation** (RAG) as the dominant architecture for ensuring data integrity [Lew+21]. RAG bridges the gap between the "parametric memory" of an LLM (what it learned during training) and the "non-parametric memory" of an enterprise (its live, changing databases). In a standard RAG setup, the system does not rely on the LLM to know facts; it uses the LLM to process facts retrieved from a trusted source [FBS25]. Without RAG, an LLM might hallucinate a product code or incorrectly merge two entities. To avoid this, before answering a query or cleaning a record, the system retrieves relevant chunks of data (from a vector database or a SQL store); these chunks serve as the "ground truth" context [Mar25]. For example, to decide if "Oracle" and "Oracle Financial Services" are the same entity, a RAG system retrieves the corporate hierarchy from an external knowledge base (like Wikidata or Dun & Bradstreet) and feeds this to the LLM [Nae+24].

One of the most novel applications of RAG in the 2024 – 2025 period is **Data Imputation** – the automated filling of missing values in dirty datasets. A proposed system, RetClean, utilizes RAG for this specific purpose [Nae+24]. When RetClean encounters a missing value (e.g., ``), it acts as a retriever.

The workflow consists of:

1. Indexing: The data lake is indexed using semantic vectors.
2. Retrieval: The system retrieves the top-k most similar complete tuples (e.g., ``).
3. Inference: It constructs a prompt containing the dirty tuple and the retrieved clean tuples, asking the LLM to infer the missing value.

RetClean is designed to handle different privacy constraints. It supports (i) using public cloud LLMs (like GPT-4) for non-sensitive data, (ii) using RAG with a local, private data lake for sensitive enterprise data, (iii) fine-tuning small, local LLMs on the retrieved data to avoid sending any data to the cloud [Nae+24].

A critical weakness of standard RAG is its reliance on Semantic Similarity (vector distance) for retrieval. Vectors are good at capturing topical relevance but can struggle with precise terminological distinctions (e.g., distinguishing "Java" the island from "Java" the programming language) [GPM25]. To solve this, researchers introduced **Entity Linking Enhanced RAG** ELERAG. This architecture integrates an Entity Linking module into the retrieval phase: it identifies entities in the query and links them to a knowledge base (like Wikidata); it then uses a **Hybrid Score Weighting Model** and **Reciprocal Rank Fusion** (RRF) to combine the results of Semantic Search (vector-based retrieval) and Entity-Based Search (Symbolic retrieval based on the linked entities). Experiments on educational datasets show that this hybrid approach significantly improves factual precision. In domain-specific contexts, the RRF hybrid outperforms both pure vector search and complex cross-encoder re-rankers, proving that symbolic grounding is essential for high-accuracy RAG [GPM25].

Finally, RAG is transforming how data transformation code itself is written. In the era of "Data Science Agents", systems are tasked with generating Python/SQL code to clean and analyze data. The main issue is that generic code LLMs (like Codex [Che+21]) do not know an enterprise's internal data schemas or utility libraries. By indexing the local codebase and documentation, RAG systems allow the agent to retrieve the specific function definitions (e.g., `clean_customer_id()`) available in the environment. This enables the generation of code that is not just syntactically correct, but functionally executable within the enterprise's specific infrastructure [Yan+25b]. Despite these aids, benchmarks like **DSBench** show that current agents still struggle with complex, multi-step data tasks, achieving pass rates of only 34-39%, indicating a significant "gap" that remains to be closed [Jin+24, Ouy+25].

2.4.5 Benchmarking Semantic Tasks: Measuring Progress

The validity of any scientific field rests on its benchmarks. In Semantic Data Fusion, the 2020 – 2025 period saw a necessary move away from small, academic datasets (like Cora, DBLP-ACM [KTR10, Orb19]) to massive, multi-dimensional benchmarks that stress-test the generalization capabilities of modern models. The **WDC Products Benchmark**, refined between 2022 and 2024, is currently the most rigorous testbed for Entity Matching [PSB24]. It is derived from the Common Crawl and contains over 11.000 product offers describing 2162 entities. Unlike

previous benchmarks, it is designed to be **Multi-Dimensional** [PDB23]. This term is adopted as the benchmark evaluates systems along three orthogonal axes:

1. Amount of Corner-Cases: The dataset intentionally includes "hard" pairs – products that are very similar but different (e.g., same brand/model, different RAM). This tests the model's ability to discern subtle semantic cues.
2. Amount of Unseen Entities: It features test sets where the entities do not appear in the training set. This tests Inductive Generalization (can the model match "Pixel 6" after only seeing "iPhone 13"?).
3. Development Set Size: It varies the amount of training data available, testing Data Efficiency.

Recent evaluations on WDC Products have yielded critical insights:

- The benchmark supports both pairwise matching (binary classification) and multi-class classification (assigning a record to an entity cluster). Models like R-SupCon (RoBERTa with Supervised Contrastive Learning) perform significantly better in the multi-class setting, achieving F1 scores 3 – 6% higher than their pairwise counterparts [PDB23].
- All supervised systems (Ditto, **HierGAT** [Wu+24])) struggle with unseen entities. However, LLM-based approaches (tested in related studies using WDC) show superior resilience here. This validates the hypothesis that "world knowledge" (pre-training) is the key to solving the OOD generalization problem [PSB24].
- The use of contrastive loss — pulling embeddings of matching records together and pushing non-matches apart — is proven to be more data-efficient than standard cross-entropy loss for EM tasks [PDB23].

For the task of generating data integration logic (code), **DSBench** represents the frontier [Jin+24, Ouy+25]. It includes 466 data analysis tasks and 74 modeling tasks sourced from Kaggle competitions and real-world scenarios. The benchmark evaluates the ability of agents to perform end-to-end tasks: understanding a user query, writing code to load/clean data, and generating a result. The results are humbling for the AI community. The best-performing model (GPT-4o) achieved a pass rate of only 39% on data generation tasks [Jin+24]. The primary failure modes include an inability to handle long contexts (large schema definitions), multi-table reasoning (complex joins), and error correction (fixing code when it fails). This highlights that while "Chat with Data" is a solved problem for simple queries, "Autonomous Data Engineering" remains an open challenge [Jin+24].

To evaluate RAG systems themselves, new frameworks like **RAGBench** (100k examples) and Microsoft's **BenchmarkQED** have been introduced [FBS25, Pot25]. These frameworks decompose evaluation into *Retrieval Quality* (*Precision@K* and *Recall@K*, they evaluate the system's ability to retrieve relevant information) and Generation Quality (*Factuality* and *Coherence*, to evaluate the possibility of hallucinations and adherence to retrieved information). Metric-wise, the community is moving beyond n-gram metrics like **ROUGE** (which measures word overlap) to semantic metrics like **BERTScore** and **LLM-as-a-Judge** (using a strong LLM to evaluate the factual consistency of a weaker model's answer) [FBS25].

CHAPTER 3

A Systematic Framework for the Evaluation of Irregularly Sampled Time Series Forecasting Workflows

The fundamental paradigm of modern TSA has long been predicated on an illusion of regularity. For decades, the dominant computational frameworks — from the Autoregressive Integrated Moving Average (ARIMA) models of the mid-20th century to the Recurrent Neural Networks (RNNs) and Convolutional Neural Networks (CNNs) that drive the current DL revolution — have operated under the implicit assumption that the world ticks in unison. In this canonical view, data points arrive at fixed, equidistant intervals Δt, forming a coherent, reliable grid that aligns seamlessly with the discrete clock cycles of digital processors [KNV23, Oh+25]. This assumption of synchronicity has allowed researchers to abstract time into a simple sequence of indices $t \in \{1, 2, ..., T\}$, treating the temporal dimension as merely an ordered list where the interval between steps is constant, irrelevant, or effectively absorbed into the model's weights. However, this abstraction is increasingly discordant with the empirical reality of modern data acquisition. As we move from carefully curated, low-frequency economic indicators to the high-velocity, event-driven streams of the Internet of Things (IoT), high-frequency finance, and electronic health records (EHR), the assumption of regularity collapses. In these domains, data is fundamentally irregular. Observations are not recorded when a clock ticks, but when an event occurs: a neuron fires, a stock trades, a sensor detects a vibration, or a clinician orders a lab test [Cha+23, KNV23]. The result is a sequence of observations where the time intervals between points are not only variable but often stochastic and highly informative. The shift from regular to irregular sampling is not merely a logistical nuisance to be solved by preprocessing; it represents a fundamental change in the underlying Data Generating Process (DGP) that necessitates a complete architectural overhaul. When Δt varies, it carries semantic information [Pat+24]. In critical care medicine, for example, the frequency of patient monitoring is often a direct proxy for patient severity – a phenomenon known as "informative sampling" or "informative missingness" [GLZ24]. Consequently, the absence of a data point is often as significant as its presence. The following sections dissect the technical hardness of modeling irregularity, exploring why standard DL inductive biases fail in this regime. The limitations of traditional "imputation-then-prediction" pipelines will be analyzed, as they destroy temporal precision and introduce dangerous artifacts [Kid+20]. Continuous-time frameworks will be discussed — specifically **Neural Ordinary Differential Equations** (Neural ODEs), **Neural Controlled Differential Equations** (Neural CDEs), and **Latent Stochastic Differential Equations** (Latent SDEs) — and the way they treat the hidden state will be further explored: they do not treat it as a discrete map but as a continuous trajectory evolved by a learned vector field [LLY23]. Finally, the benchmarking ecosystem will be critically evaluated, exposing the methodological flaws in current "fixed origin" evaluation protocols that inflate performance metrics and obscure the true robustness of these models [Kid+20, Li+20b, RCD19].

3.1 The Technical Challenges of Irregularity

The "inductive bias" of an ML algorithm refers to the set of assumptions the learner makes to predict outputs for unseen inputs. These biases are essential for generalization; they allow a model to learn from finite data by constraining the hypothesis space. However, the inductive biases that make CNNs and RNNs so powerful on regular data — translation invariance and sequential order — become liabilities when applied to ISTS.

In a standard 1D CNN, a kernel slides over the input sequence with a fixed stride. The mathematical operation of convolution assumes that the spatial or temporal distance between input features is constant. Specifically, the network assumes that the relationship between input x_t and $x_{\{t+1\}}$ is structurally identical to the relationship between $x_{\{t+1\}}$ and $x_{\{t+2\}}$. In an irregularly sampled series, this is patently false. Consider a patient whose heart rate is measured at 9:00 AM (t_1), 9:15 AM (t_2), and then not until 6:00 PM (t_3). A standard CNN processing this sequence as a vector $[x_1, x_2, x_3]$ essentially interprets the 15-minute gap between the first two points as equivalent to the nearly 9-hour gap between the second and third [KNV23]. **This temporal distortion prevents the model from correctly learning the rate of change (dynamics) of the system**. If the heart rate changes by 10 bpm between t_1 and t_2, it represents a rapid acceleration. If the same change occurs between t_2 and t_3, it represents a slow drift. A standard CNN, blind to the timestamps, cannot distinguish between these two radically different physiological states. Recent work on **Time-Parameterized CNNs** (TPCNNs) attempts to mitigate this by parameterizing kernels as continuous functions of time, but the underlying rigidity of the convolutional grid remains a significant hurdle [KNV23]. Similarly, **Recurrent Neural Networks** (RNNs) update their hidden state h_t using a recurrence relation $h_{\{t\}} = \sigma\Big(W_h h_{\{t-1\}} + W_x x_t + b\Big)$. This formula implicitly assumes that the transition from $t-1$ to t represents a uniform "step" in time. The weights W_h encode the dynamics of the system over this fixed interval. When the interval Δt varies, the dynamics governing the system's evolution also change. In a physical system governed by diffusion or decay, the state at time $t + \Delta t$ is a function of Δt (e.g., $e^{\{-\lambda \Delta t\}}$). A standard RNN has no mechanism to adjust its weights W_h based on the duration of the gap, forcing it to learn an "average" transition that fits neither short nor long intervals well [Kid+20, RCD19]. To mathematically accommodate this, the recurrence must become time-variant. This realization led to early adaptations like the **Phased LSTM**, which introduced a time gate oscillating with a learnable frequency, and eventually to the **GRU-D**, which explicitly decays the hidden state based on the observed Δt [Tan+20, Zha+24a].

However, these are discrete patches on a fundamentally continuous problem. Beyond the mathematical/theoretical issues, irregularity introduces severe systems-level engineering challenges. DL hardware (GPUs/TPUs) is highly optimized for dense, synchronous matrix operations. A batch of regular TS can be represented as a dense tensor of shape `(Batch_Size, Sequence_Length, Features)`. Irregular TS, however, manifest as "ragged tensors" or variable-length sequences where events do not align across samples in the batch. The naive solution is to pad all sequences to the length of the longest sequence in the batch. In domains like healthcare, where one patient might have 10 observations and another 10.000. this results in sparse matrices that are 99% padding, wasting vast amounts of GPU memory and compute cycles on multiplying zeros [Feg+22, Ran25]. Advanced frameworks attempt to "pack" these sequences or use gather/scatter operations to process only valid data points. However, this breaks the memory coalescing patterns that GPUs rely on for speed, leading to significant latency penalties. Some researchers have proposed abandoning the sequence structure entirely in favor of set-based architectures. The SeFT (**Set Functions for Time Series**) framework, for instance, treats the input not as a

sequence but as an unordered set of observation triplets $\{(t_i, x_i, m_i)\}$. By using permutation-invariant aggregation functions (like Sum or Mean pooling after a neural embedding), SeFT bypasses the ragged tensor problem and is highly parallelizable [Hor+20]. However, this approach sacrifices the explicit sequential inductive bias, which can be detrimental for tasks where the order of events is causally critical (e.g., drug administration followed by a reaction).

Confronted with the difficulty of modeling irregularity directly, a common strategy in the applied literature (2015 – 2020) was "imputation-then-prediction". This involves defining a fixed reference grid (e.g., hourly) and mapping the irregular observations onto this grid using methods like Last Observation Carried Forward (LOCF), mean imputation, or linear interpolation [LZ21]. While computationally convenient — it converts the ISTS problem back into a regular time series problem solvable with standard tools — this approach is fraught with danger:

- Binning forces events to snap to the nearest grid point. A drug given at 10:59 and a reaction at 11:01 might be binned into the same "11:00" bucket, destroying the causal precedence information required for the model to learn that the drug caused the reaction. Conversely, they might be separated into different hours, exaggerating the latency [Cha+23].
- Linear interpolation assumes a smooth, constant rate of change between points. In volatile financial markets or unstable physiological states, this assumption acts as a low-pass filter, smoothing out the very spikes and variance that are predictive of a crash or cardiac arrest [LAC17]. The model learns the properties of the interpolant, not the underlying system.
- If the data is **missing not at random** (MNAR), imputation biases the feature distribution. For example, if temperature is only measured when a patient has a fever, the observed temperatures will be high. If we impute the gaps with the population mean (normal temperature), we are fabricating data that contradicts the patient's actual (likely febrile) state during the unobserved intervals [Per+22, SM19].

The consensus in the recent literature (2021-2025) is that imputation-free or end-to-end architectures, which ingest the raw irregular triplets (t, x, m) directly, are generally favored over pipeline approaches [Liu+25c, Neo+25].

3.2 The Taxonomy of Missingness: Mechanisms and Implications

To build robust models for ISTS, one must understand the generative process of the "missingness" itself. The statistical literature, grounded in Rubin's seminal theory (1976), categorizes missing data mechanisms into three distinct classes. In the context of TS, these definitions take on temporal nuances that are critical for forecasting [Emm+21, RUB76].

3.2.1 Missing Completely at Random (MCAR)

Under the MCAR assumption, the probability of a data point being missing is independent of both the observed data (X_{obs}) and the unobserved data (X_{miss}):

$$P(R|X_{\text{obs}}, X_{\text{miss}}) = P(R) \tag{3.1}$$

where R is the missingness indicator matrix. In an IoT sensor network, this might correspond to random packet loss due to atmospheric interference or a battery dying unpredictably. In a clinical trial, it might be a sample lost in transit. If the data is MCAR, the observed data is a representative random sample of the complete data. Simple deletion of incomplete rows or mean imputation yields unbiased (though less efficient) estimates. However, in most ISTS domains — especially healthcare — MCAR is a vanishingly rare condition. Assuming MCAR when it does not hold leads to catastrophic underestimation of risk.

3.2.2 Missing at Random (MAR)

Under MAR, the missingness is systematic but can be fully explained by the observed data:

$$P(R|X_{\text{obs}}, X_{\text{miss}}) = P(R|X_{\text{obs}}) \tag{3.2}$$

An example of this would be the following scenario: a physician orders a creatinine blood test (X_{miss}) specifically because the patient's previously measured blood pressure (X_{obs}) was elevated. The missingness of creatinine is not random, but its probability is conditional on variables we do have. This is the domain of sophisticated imputation techniques like **M**ultivariate **I**mputation by **C**hained **E**quations (MICE) or **attention-based imputation** mechanisms. If the model is conditioned on the observed history, it can theoretically infer the likely values of the missing data without bias [ZAB24].

3.2.3 Missing Not at Random (MNAR) and Informative Sampling

The most challenging and common scenario in ISTS is MNAR, where the missingness depends on the value of the unobserved variable itself:

$$P(R|X_{\text{obs}}, X_{\text{miss}}) = P(R|X_{\text{miss}}) \tag{3.3}$$

An example: a patient feels perfectly healthy and decides to skip a scheduled check-up. The blood pressure measurement is missing because it is normal. Conversely, in the ICU, a patient is deteriorating, so the nurse increases the monitoring frequency from hourly to every 15 minutes. The presence of dense data points is caused by the extreme values (high severity) of those points [GLZ24, LKW]. This phenomenon is often termed "informative sampling" or "informative missingness". The timing of the observations contains a "leak" about the hidden state. If a model ignores this and simply imputes the mean (normal) value during a gap, it misses the signal that the gap itself implies stability (or, in other contexts, neglect). Conversely, high-frequency bursts of data should be treated as a feature indicating high variance or criticality [Gro20, Pat+24]. Recognizing that the missingness mask M and the time intervals Δt are highly predictive features, modern architectures explicitly encode them. Lipton et al. [LKW] in 2016 demonstrated that

simply augmenting the input vector x_t with a binary vector m_t indicating which variables are measured significantly improved diagnostic performance, often surpassing models that only had access to the values. In some cases, the information on what tests a doctor ordered was more predictive of the diagnosis than the results of the tests [LKW, SSO21]. The time interval Δt since the last observation is also fed as a direct input. This allows the network to learn context-specific decay rates. For example, a high glucose reading might be relevant for hours, while a high heart rate reading might become stale in minutes. By learning weights associated with Δt, the model can adapt its memory horizon dynamically for each channel [Tan+20, Wan+24b].

3.3 Architectural Frameworks: The Shift to Continuous Time

The inadequacy of discrete RNNs and the artifacts of interpolation have driven the field toward Continuous-Depth Deep Learning. This paradigm defines the hidden state evolution using differential equations, allowing the model to operate in continuous time natively.

Before fully continuous models matured, researchers developed specialized RNN cells to handle irregularity. The **GRU-D** architecture [Che+16] modifies the standard GRU cell to explicitly model the "staleness" of missing information. It introduces a decay term γ calculated from the time delta Δt:

$$\gamma_t = \exp\big(-\max\big(0.W_\gamma \Delta t + b_\gamma\big)\big) \tag{3.4}$$

The hidden state h_{t-1} is decayed toward zero (or a learned baseline) before being used to update the current state:

$$\hat{h}_{t-1} = \gamma_t \odot h_{t-1} + (1 - \gamma_t) \odot \text{baseline} \tag{3.5}$$

This formulation mathematically enforces the intuition that information degrades over time. Crucially, the decay rates W_γ are learned per-variable. The model might learn that blood pressure measurements "decay" slowly (stay relevant longer) while heart rate measurements decay quickly. GRU-D was a pivotal step, demonstrating that baking temporal physics into the architecture outperforms black-box RNNs on imputed data [Tan+20].

The **Phased LSTM** [NPL16] architecture addresses irregularity from a signal processing perspective. It adds a time gate k_t controlled by a rhythmic oscillation with period τ, phase s, and shift ratio $r_{\{\text{on}\}}$. The gate opens only during a small percentage of the cycle. This allows the LSTM to update its state only when the time t aligns with its internal rhythm. This effectively acts as a learnable sampling rate, allowing the network to process inputs at different timescales (e.g., minute-by-minute vs. daily) and ignore high-frequency jitter in irregularly sampled bursts [Zha+24a].

The introduction of Neural ODEs marked a shift toward continuous-time frameworks in ISTS forecasting [HP20, Kid+20, RCD19]. Instead of a discrete sequence of layers or steps, the hidden state $z(t)$ is defined as the solution to an Initial Value Problem (IVP):

$$\mathrm{d}\frac{z(t)}{\mathrm{d}}t = f_\theta(z(t), t, x) \tag{3.6}$$

where f_θ is a neural network parameterizing the derivative (the vector field). To find the state at any future time t_i, a numerical Ordinary Differential Equation (ODE) solver is used (e.g., Runge-Kutta, Dormand-Prince):

$$z(t_i) = z(t_0) + \int_{t_0}^{t_i} f_\theta(z(t), t, x)\,\mathrm{d}t \tag{3.7}$$

This method enables a series of advantages for ISTS processing:

- The solver can define the state at any continuous time point. The model naturally "glides" over gaps in the data, evolving the state according to the learned physics of the system. There is no need for imputation; the gap is simply a longer integration interval [RCD19].
- The dynamics function f_θ is shared across all time, distinguishing it from deep networks where each layer has distinct weights.
- Training is performed using the **adjoint method**, which solves an auxiliary ODE backward in time to compute gradients. This allows training with constant memory cost with respect to the number of integration steps, a massive advantage for long sequences [Kid+20, Li+20b].

Neural ODE presents interesting architectural variants:

1. Latent ODEs: These use an encoder (often an ODE-RNN running backward) to compress the entire irregular TS into a single latent vector z_0. A generative ODE then reconstructs the series forward in time. This is particularly powerful for interpolation and unsupervised learning tasks [RCD19].
2. ODE-RNN: This hybrid model combines the best of both worlds. It updates the hidden state discretely when an observation arrives (using a standard RNN update to incorporate new information) but evolves the state continuously between observations using an ODE. This captures the instantaneous "jump" of new data and the continuous "flow" of the system dynamics [RCD19].

A limitation of Neural ODEs is that the trajectory is determined entirely by the initial condition z_0 and the autonomous dynamics. Adjusting the trajectory based on incoming data requires interrupting the solver and resetting the state (as in ODE-RNNs). **Neural Controlled Differential Equations** (Neural CDE) [Jhi+23, Kid+20] offer a more mathematically elegant solution by defining the evolution as driven by the continuous path of the data $X(t)$ itself:

$$\mathrm{d}z(t) = f_\theta(z(t))\,\mathrm{d}X(t) \tag{3.8}$$

This is a Riemann-Stieltjes integral. The discrete data points are first interpolated into a continuous path $X(t)$ (often using cubic splines), and this path serves as the control signal. Neural CDEs allow the hidden state to react continuously to the input stream, and this enables them to be superior for processing incoming data streams where the "response" to an input is continuous and prolonged [Jhi+23].

Deterministic ODEs assume that the future is a fixed function of the past. However, real-world systems (especially biological and financial) are stochastic. There may be unobserved forces (confounders) acting on the system. **Latent Stochastic Differential Equations** (Latent SDEs) [FGS25, Li+20b] extend the framework by adding a diffusion term:

$$\mathrm{d}z(t) = \underbrace{f_\theta(z(t), t)\,\mathrm{d}t}_{\text{Drift}} + \underbrace{\sigma_\theta(z(t), t)\,\mathrm{d}W(t)}_{\text{Diffusion}} \tag{3.9}$$

where $W(t)$ is a Wiener process (Brownian motion). The diffusion term causes the variance of the latent state to grow over time. During long gaps in data, the model's confidence naturally decreases (variance increases). This provides a rigorous, built-in measure of predictive uncertainty, which is critical for decision-making [Li24, ZGK23]. However, training SDEs is significantly harder than ODEs. It requires stochastic adjoint methods (which must handle the non-differentiability of Brownian motion) and careful choices between Itô and Stratonovich integration schemes [Den+21, Li+20b].

Table 3.1 synthesizes the capabilities of these frameworks, their pros, and their cons from a point of view of the capability of the method to handle irregularity, the robustness to missingness, and the computational complexity. Under these points of view, all evidence points towards the adoption of SeFT as a globally capable methodology: it has a low computational cost, and it is highly robust to missingness; however, as previously highlighted, treating sequences as sets doesn't allow SeFT to perform at its best in scenarios where the exact sequential transition is more important than the global distribution of values; in this case, recurrent models like GRU-D or continuous models like Neural ODEs are often superior in capturing local dynamics. Moreover, even though it handles the missingness, SeFT is not able to meaningfully interpret the absence of data as a feature. Finally, Set Aggregation can be detrimental when there are very long-range dependencies and the relative distance between points is critical.

Architecture	Core Mechanism	Handling of Irregularity	Robustness to Missingness	Computational Complexity	Best Use Case
Standard LSTM/GRU + Imputation	Discrete recurrence on fixed grid	Requires preprocessing (binning/ imputation)	Low: Artifacts from imputation propagate errors	Low: Fast training, optimized kernels	Baselines, dense data with few gaps.
GRU-D	Discrete recurrence with learned decay	Explicit decay $\exp(-\gamma \Delta t)$	Medium: Models "staleness" of data	Low-Medium: Adds simple operations to GRU cell	EHR mortality prediction, informative missingness.
Neural ODE	Continuous vector field (IVP)	Native: Solves $z(t)$ for any t	High: Glides over gaps via dynamics	High: ODE solver overhead, slow training (adjoint)	Physical systems, sparse irregular data
Neural CDE	Controlled differential equation (integral)	Continuous control by data path $X(t)$	High: Adapts to incoming stream continuously	High: Cubic spline interpolation + solver	Real-time streams, high-frequency finance
Latent SDE	Stochastic dynamics (diffusion)	Native continuous + noise process	Very High: Captures uncertainty of trajectory	Very High: Stochastic adjoint, complex convergence	Uncertainty quantification, biological systems
SeFT	Set Aggregation / Attention	Treats sequence as a set $\{(t, x)\}$	High: Invariant to order/gaps	Low: Highly parallelizable, no recurrence	Large-scale EHR classification, parallel processing

Table 3.1: Comparative Summary of Forecasting Architectures.

3.3.1 The Benchmarking Landscape

The development of sophisticated models has, unfortunately, outpaced the rigor of the benchmarks used to evaluate them. The current landscape is dominated by a few datasets and evaluation protocols that are increasingly viewed as flawed.

The MIMIC-III (Medical Information Mart for Intensive Care) database is the ubiquitous benchmark for ISTS research [Joh+16]. While invaluable, its overuse has created a "monocul-

ture": MIMIC-III data comes from a single medical center (Beth Israel Deaconess Medical Center) in Boston; it reflects the specific demographics, care protocols, and administrative coding practices of that institution. Models optimized for MIMIC-III often fail to generalize to other hospitals ("dataset shift"). The standard tasks — mortality prediction and length of stay (LOS) — have been "solved" to a degree where marginal gains are often due to overfitting or subtle data leakage rather than architectural superiority [Har+19, RBH22].

Perhaps the most critical methodological failure in the field is the prevalence of **Fixed Origin Evaluation** (also known as "fixed holdout"). In this setup, researchers split the time series at a static point T (e.g., using the first 24 hours of ICU admission to predict mortality). The model is trained and tested on this single snapshot. This ignores the temporal reality of the data. In a real clinical setting, predictions are required continuously – at hour 2, hour 5, hour 48, etc. Fixed Origin Evaluation encourages models that overfit to specific artifacts present at a fixed time point, e.g., $t = 24$. It fails to penalize unstable models: a model might be excellent at hour 24 but terrible at hour 25. It also fails to evaluate how well the model updates its belief as new data arrives [SPA23, Zan25]. The solution is **Rolling Origin Evaluation**: a rigorous benchmark must use Rolling Origin (or **Prequential**) evaluation, or a similar approach thereof. Here, the forecast origin t moves forward through the Time Series. The model makes a prediction at t, receives the true value, updates its state (or is retrained), and predicts at $t + 1$. This effectively generates a distribution of errors over time, providing a true measure of the model's reliability in deployment [Zan25]. Papers that do not use rolling origin evaluation are to be considered biased in the modern context. To address these gaps, the community is developing more rigorous benchmark suites: **Physiome-ODE** and **Time Series Forecasting Benchmark** (TFB). The former introduces the **Joint Gradient Deviation** (JGD) score to quantify the "stiffness" and difficulty of the time series dynamics. It explicitly selects datasets that challenge the continuous integration capabilities of ODE-based models, filtering out trivial linear trends [Klö+25]. The latter is an automated pipeline covering 10 diverse domains (traffic, energy, environment, economics, etc.). TFB aims to eliminate "stereotype bias" (where DL papers pick datasets where DL wins and ignore those where statistical methods dominate) by enforcing a wide, diverse evaluation protocol [Qiu+24].

3.3.2 Sensitivity and Robustness Evaluation

In high-stakes domains like healthcare, robustness — the stability of the model under perturbation — is often more important than raw accuracy. A model that predicts mortality with 95% accuracy but flips its prediction if a single blood pressure measurement is dropped is clinically dangerous. To quantify robustness, researchers employ **Stochastic Corruption**. This involves artificially degrading the test data by masking observed values (simulating sensor failure or data transmission loss) and measuring the performance drop. Frameworks like RMD (**Randomized Missing Data**) formalize this [DHS22, DS25]. The model is tested at varying levels of missingness (e.g., 10%, 20%, ... 90%); the result is then visualized as a degradation curve plotting Accuracy vs. Missing Rate. Robust models (like Latent SDEs and SeFT) typically exhibit a "convex" curve, maintaining high performance even as data becomes sparse, while fragile models (like standard LSTMs with imputation) show precipitous "concave" drops [Li24, TCK25]. RMD analysis reveals a **bias-variance trade-off** controlled by the missingness rate. Including "noisy" or "outlier" measurements can sometimes harm performance. Robust models effectively perform "randomized inattention", learning to ignore unreliable data points [DHS22].

The "Lucky Seed" Phenomenon. Recent large-scale benchmarking has revealed that the variance in performance due to random seed initialization (which affects weight initialization and data shuffling) can often exceed the performance gains claimed by novel architectures. It is now a standard requirement for rigorous research to report the mean and standard deviation of metrics over at least 5-10 random seeds. Papers reporting only a single "best" run are statistically invalid in this domain [KM25, Lu+25].

Decoupling Interpolation from Forecasting. A critical nuance is that Interpolation Accuracy (MSE of filling gaps) does not strictly correlate with Downstream Task Accuracy (AUC of mortality prediction). In cases of informative missingness (MNAR), **an accurate interpolation might be harmful**. If a patient is stable, their vitals are not measured. An interpolation model might correctly guess the "normal" value for the missing timestamp. However, by filling in that "normal" value, it might obscure the fact that the absence of the measurement was the strongest predictor of stability. Benchmarks must evaluate both Reconstruction Error (MSE) and Downstream Performance (AUC/F1) independently. Optimizing for MSE alone is often a misaligned objective in informative sampling regimes [Cha+23, Wan+24b].

3.3.3 Gap Analysis and Proposed Solutions

While Neural ODEs offer "physical" dynamics, the learned vector field f_θ is usually a black-box neural network. It does not provide the explicit equations (e.g., $\frac{dx}{dt} = -kx$) that scientists desire. Future work must integrate Symbolic Regression into the Neural ODE loop. Techniques that sparsify the neural network weights to recover governing equations (e.g., SINDy combined with Neural ODEs) are a promising frontier for making these models trustworthy and scientifically discoverable [WJG24].

Current models excel at numerical time series but struggle to integrate asynchronous multi-modal data (e.g., clinical notes, static demographics, and imaging) into the continuous flow: a note written at $t = 5.3$ should influence the continuous trajectory of the patient state. Hybrid architectures that fuse Transformers (for text/static data) with Neural CDEs (for time series) address this. The static/text embeddings can serve as the "control" signal or conditioning vector for the differential equation, allowing discrete semantic events to modulate continuous physiological dynamics [Cha+25, Gru+24].

While LLMs have revolutionized NLP, a "Foundation Model for Time Series" remains elusive. The main hurdle is tokenization. How do you tokenize continuous, irregular time? The field is exploring **Large Time Series Models** (LTSMs) that pre-train on massive corpora of diverse time series using self-supervised objectives (like masking). Recent work suggests that LLMs (like GPT-4) can perform zero-shot forecasting by treating time series as strings of text digits, but they struggle with arithmetic precision and long horizons [Gru+24]. The path forward likely involves **Patching** (breaking series into sub-sequences) and specialized embeddings that encode continuous time directly into the transformer's attention mechanism [LLY23, MPW25].

To address these combined challenges of technical friction, fragmented evaluation, and the `No Free Lunch` theorem, this chapter introduces A Systematic Framework for the Evaluation of Irregularly Sampled Time Series Forecasting. The primary contribution is a high-heterogeneity benchmarking environment, designed to assess model robustness across multiple dimensions. Specifically, the framework aims to mitigate dependence on specific missingness patterns through

a stochastic corruption strategy (multiple seeds) and simultaneously attempts to address the Fixed Origin Fallacy thanks to this technique. The remainder of this chapter is structured as follows: Section 3.4 details the architectural design of the framework; Section 3.5.1 and Section 3.5.2 describe the heterogeneous selection of datasets and models under comparison; Section 3.6 details the rigor of the evaluation metrics; Section 3.8 presents and discusses the core results and implications for future ISTS research.

3.4 ISTSBenchmark Framework Architecture and Design

The primary objective of the proposed framework is to provide a standardized and extensible environment for the rigorous evaluation of forecasting models across a diverse array of Irregularly Sampled Time Series (ISTS) scenarios. Recognizing the fragmentation in current benchmarking practices, this architecture is built upon the principles of modularity, reproducibility, and cross-domain generalizability.

3.4.1 Conceptual Design and Objectives

The framework is designed to address the multifaceted nature of the "No Free Lunch" theorem in temporal domains, which postulates that no single architecture is universally superior across all data distributions. To investigate this, the architecture facilitates a "wide-vision" analysis along three primary axes:

1. Domain Heterogeneity: Evaluating performance across disparate fields — ranging from high-frequency telemetry to sparse social or physical systems — to determine if specific architectural inductive biases are better suited for particular semantics.
2. Missingness Robustness: Systematically assessing how various missingness patterns and sampling irregularities influence model stability.
3. Methodological Comparative Analysis: Providing an objective "head-to-head" comparison between DL models, specifically imputation-based pipelines (Imputation → Standard Forecasting) and ISTS-specific models (Preprocessing → Missingness-handling Forecasting).

3.4.2 Modular Pipeline Architecture

A core contribution of this work is the development of a maintainable and efficient pipeline that abstracts the complexity of ISTS forecasting. The framework is structured into four decoupled layers:

1. Data Ingestion: This layer manages raw data loading, applying the desired normalization strategy (standardization, by default, normalization, or none). A core innovation of this framework is the systematic handling of **multi-source** datasets (e.g., USHCN, FrenchPiezo), a term created to refer to datasets where multiple sources contribute to the data. Unlike standard multivariate time series, where all variables are observed

simultaneously on a shared grid, multi-source data involves independent samples that may span differing temporal ranges. This architecture implements per-sample reindexing and resampling to ensure a unified temporal window, a process that is significantly more computationally expensive but necessary for real-world sensor networks.

2. Corruption Layer: Applies the stochastic corruption strategy. By utilizing multiple random seeds to "dirty" the datasets, the framework generates various realizations of missingness, ensuring that the resulting benchmarks are robust and not artifacts of a single specific gap pattern.

3. Imputation Layer: The corruption needs to be filled using some strategy for the standard forecasting models to work correctly. The imputation layer addresses the issue by applying the user-requested methodology over **non-overlapping** sliding windows. The imputation method can be simple, i.e., LOCF, Mean, and similar methods, or a DL model taken from the recently published library **PyPOTS** [Du+24, Du+25]. To maintain fine-grained control over the process, the models have been taken out of their pre-made training framework; various instances of the same model are created and trained with different hyperparameters (taken from the original paper that introduced PyPOTS) through a grid search process on different look-back window sizes for the same dataset. This guarantees the best imputation performance for that model. If the model is an ISTS-specific one, this step is completely skipped.

4. Forecasting Layer: To combat the Fixed Origin Fallacy, the framework employs an **overlapping** sliding window approach for training data generation. By shuffling these temporal windows during the training phase, the model is exposed to a wide variety of temporal regimes, enhancing its ability to capture dynamic patterns without being anchored to a single historical point.

The framework's architecture is designed for extensibility, allowing researchers to easily integrate new models or datasets. This ensures that as the state-of-the-art evolves, the framework remains a relevant and flexible tool for the community – as long as the process involves a "training → validation → test" procedure. The train, validation, and test functions can be overridden to adapt the process ad hoc per model, enabling the usage of the framework for non-DL models.

In case of spatially expensive datasets, the framework provides the possibility to exploit NVIDIA RAPIDS' [RAP] technology to accelerate data handling through the GPU; it's currently optimized for future high-velocity deployments and wasn't needed for the kind of dataset used for the preliminary tests. The usage of the GutenTag library [WSP22] was also considered and stubbed for synthetic time series generation to enable controlled experiments with specific mathematical properties; however, the benchmark focuses on well-known datasets used in literature.

Finally, the framework treats forecasting as a multi-step engineering challenge. It investigates the impact of the forecasting horizon on model outcome, identifying the specific "break points" where certain architectures (e.g., Transformers vs. Linear models) begin to fail or excel. By keeping the preprocessing steps constant across all models, the framework ensures that performance variances are attributable solely to the architectural design or the chosen imputation technique, rather than disparate data handling.

3.5 Experimental Setup

The framework evaluates a broad spectrum of architectures to capture the evolution of TSA and the specific challenges of irregular sampling. The models are categorized based on their structural approach to temporal modeling and their handling of missingness.

3.5.1 Dataset Taxonomy

Table 3.2 compares the different datasets adopted in the framework, describing their characteristics relative to their domain, data type, frequency, and whether they are multi-source or not.

Dataset	Domain	Type	Frequency	Multi-Source
USHCN	Climate	Environmental	Daily	Yes
FrenchPiezo	Geospatial	Environmental	Daily	Yes
BeijingPM2.5	Air Quality	Environmental	Hourly	Yes / No
ETTh1 / ETTh2	Energy	Infrastructure	Hourly	No
ETTm1 / ETTm2	Energy	Infrastructure	15-min	No
Electricity	Energy	Infrastructure	Hourly	No
ExchangeRate	Finance	Economic	Daily	No
Weather	Climate	Environmental	10-min	No
ILI	Health	Epidemiological	Weekly	No
AppliancesEnergy	Energy	IoT/Home	10-min	No

Table 3.2: Dataset Taxonomy.

The twelve datasets selected for this benchmark were chosen to represent a broad spectrum of temporal dynamics, ranging from high-frequency physical sensors to sparse, irregularly reported epidemiological data. A defining feature of this selection is the inclusion of multi-source datasets, which require independent per-sample reindexing and temporal alignment.

- USHCN (U.S. Historical Climatology Network): A daily climate dataset spanning over a century, frequently used to test long-term trend extraction.
- FrenchPiezo: A geospatial dataset of piezometric head levels, characterized by highly irregular sampling intervals and complex subterranean dynamics.
- BeijingPM2.5: Hourly air quality data from Beijing; used in both its multi-source (multiple station) and single-source variants to test spatial-temporal integration.
- ETT (Electricity Transformer Temperature): A standard benchmark in long-term forecasting (ETTh1, ETTh2, ETTm1, ETTm2), providing high-resolution infrastructure telemetry.
- ExchangeRate: Daily economic data characterized by a low signal-to-noise ratio, serving as a stress test for model overfitting.
- ILI (Influenza-Like Illness): Weekly reported health data from the CDC, representing sparse and seasonal human-centric time series.

- Weather: A 10-minute resolution dataset of local climatological variables, testing non-linear relationship modeling.
- AppliancesEnergy: IoT scenario composed of 10-minute resolution telemetry from a house in Stramsund, Norway, monitoring appliance energy use alongside environmental conditions via a ZigBee wireless sensor network.

3.5.2 Model Taxonomy

To ensure a robust comparison, representative models from several "evolutionary branches" of the field were selected:

1. Native ISTS Models: Architectures characterized by their ability to process irregularly sampled sequences directly, circumventing the need for an external imputation layer. A few examples of such architectures are GRU-D, mTAN (Multi-Time Attention Network) and PrimeNet [Che+16, Cho+23, LJD19].
2. Classical Deep Learning (RNN/CNN): Standard recurrent and convolutional baselines, including LSTM, DeepAR (probabilistic RNN), and TCN (Temporal Convolutional Network) [Lea+16, SFG19, VKB21].
3. Transformer-Based Models: State-of-the-art attention mechanisms including PatchTST, FEDFormer (frequency-domain attention), and the standard TFTransformer [Lim+20, Nie+23a, Zho+22].
4. Linear and MLP-Based: The "Linear Counter-Revolution" models, DLinear and NHITS, which prioritize trend extrapolation and simplicity over attention complexity [Cha+22, Zen+22].
5. Specialized Architectures: StemGNN, which explore graph-based and multi-scale temporal dependencies [Cao+21].

Table 3.3 summarizes the architecture and main characteristics of each model.

Category	Models	Key Architectural Focus
Native ISTS	GRU-D, mTAN, PrimeNet	Explicit Δt modeling, learned decay, and relational information propagation
Transformer	PatchTST, FEDFormer, TFTransformer	Long-range dependencies and frequency-domain attention
Linear / MLP	DLinear, NHITS	Trend extrapolation and decomposition
Classical DL	LSTM, DeepAR, TCN	Sequential memory and dilated convolutions
Specialized	StemGNN	Graph-based and multi-scale temporal dependencies

Table 3.3: Dataset Taxonomy.

The model selection spans the phylogenetic branches of Time Series Analysis, from classical recurrence to the latest techniques in DL, like the patching mechanism used by PatchTST [Nie+23b]. Each model was selected to evaluate a specific architectural inductive bias in the context of irregular sampling.

- Native ISTS Models: GRU-D introduces a decay mechanism to model the "staleness" of missing observations, mTAN utilizes a multi-time attention mechanism to learn continuous-time representations, while PrimeNet leverages relational propagation to maintain stability and informativeness across non-aligned multivariate streams [Che+16, Cho+23, LJD19].
- Transformer-Based Models: PatchTST utilizes sub-series patching to enhance the attention mechanism's local semantic awareness. FEDFormer operates in the frequency domain to capture global seasonal patterns, and the TFTransformer serves as the canonical self-attention baseline [Nie+23b, Zho+22].
- Linear & MLP Models: DLinear and NHITS utilize simple linear mappings and multi-rate sampling to extrapolate trends, challenging the necessity of complex attention layers [Cha+22, Zen+22].
- Classical & Specialized DL: DeepAR provides a probabilistic RNN approach, while TCN (Temporal Convolutional Networks) uses dilated convolutions for long-range memory. StemGNN and PrimeNet represent specialized attempts to model multi-scale and graph-based dependencies [Cao+21, Lea+16, SFG19].

Each model has its own set of original hyperparameters taken from the respective original paper. To be able to handle each configuration, all values are saved in a JSON file and then subsequently parsed. The file can be manually edited, or the hyperparameters can be overridden by passing the hyperparameter name as a CLI argument. Hypothetically, one could fix a dataset configuration and test the same model with a variety of hyperparameters changing for each run so to observe the impact it has on the forecasting performance.

3.5.3 Defining the Forecasting Scenario: Long-Term Multivariate Single-Point

A critical distinction of this benchmark is its focus on **long-term multivariate single-point forecasting**. Unlike short-term horizons where local noise dominates, long-term forecasting requires the model to capture deep structural patterns. Furthermore, while many frameworks default to "sequence-to-sequence" metrics (averaging error over the entire forecast window), our framework isolates the single-point prediction at the end of the horizon. This scenario is particularly demanding for ISTS, as the accumulation of uncertainty over irregular gaps makes the final point prediction highly sensitive to model stability.

3.5.4 The Darts Backend Redesign: Architectural Lessons

Initially, the framework was conceptualized using the Darts library as a backend [Her+22]. However, an architectural mismatch was identified: Darts focuses on sequence-wide metrics, calculating performance over the entire forecast horizon rather than a specific terminal point. To achieve the fine-grained control required for single-point evaluation and to support the custom data-loading requirements of multi-source datasets, the framework was completely redesigned. This custom architecture allows for:

- Metric Precision: Evaluating performance strictly on the target time step $t + h$.
- Multi-source Alignment: Overcoming the limitations of standard dataloaders to manage samples with independent temporal ranges.
- Extensibility: Maintaining "stubs" for future integration of models like **XGBoost** [CG16], **ARIMA** [Wil16], **CRU** [SKB24], and **eGRU** [Zha+24b], ensuring the framework serves as a long-term research tool.

3.6 Evaluation Metrics

To avoid the "SOTA chase" built on flawed benchmarks, this framework employs a multi-metric approach focused on both accuracy and reliability. Unlike sequence-to-sequence libraries (e.g., Darts), the metrics are computed **strictly on the terminal point** of the long-term horizon $(t + h)$ to assess terminal stability. The utilized metrics are:

1. MAE (Mean Absolute Error)
2. MSE (Mean Squared Error)
3. MDAE (Median Absolute Error)
4. SMAPE (Symmetric Mean Absolute Percentage Error)
5. R^2 Score (Coefficient of Determination)

MAE, MSE, and MDAE are used as primary indicators of point-accuracy, while SMAPE is included to provide scale-independent comparisons across the heterogeneous datasets. The R^2 score, or coefficient of determination, is integrated into this framework to provide a measure

of explanatory power. While absolute error metrics (MAE/MSE) quantify the distance from ground truth, the R^2 score determines whether the model's performance is derived from genuine generalization of the underlying temporal dynamics or if it is failing to outperform a naive mean-based predictor. A negative R^2 score is a critical diagnostic signal in this benchmark, highlighting instances where architectural complexity or poor imputation results in a model that performs worse than the historical average of the series.

Results in this thesis are reported as an average across multiple random seeds, quantifying the variance introduced by the stochastic corruption strategy and ensuring that "luck" in missingness patterns is minimized.

3.6.1 Procedural Methodology

Hardware Specifics. The proposed framework was developed and tested on an NVIDIA DGX platform, having 2 AMD EPYC 7702 CPUs having 64 cores each at a frequency of 2GHz, 2TB of RAM, and 8 NVIDIA A100 with 40 GBs of VRAM each.

Software Stack. The program was developed entirely in Python. It requires a virtual environment, conda environment, or similar technology to install all necessary packages, e.g., PyTorch, Pandas, Numpy, PyTorch, Scikit-Learn. As a future-ready feature, the usage of NVIDIA RAPIDS is also implemented but not thoroughly tested.

The framework is designed as a modular orchestration system that prioritizes experimental reproducibility and computational efficiency. By decoupling the data ingestion, corruption, imputation, and forecasting layers, the system ensures that performance variances are strictly attributable to architectural differences rather than inconsistent preprocessing.

The software utilizes a centralized orchestration module to manage the experimental workload through a Cartesian product of parameters. This approach facilitates a comprehensive "wide-vision" analysis, where one or more configurations (models, datasets, missingness patterns, and forecasting horizons) are evaluated in parallel. The user can provide one or more of:

- Models to be trained for the forecasting phase;
- Filling techniques to clean the datasets from missing values (simple methods or DL models)
- Missing data percentages (in the $[0, 1]$ range) to be applied to the dataset;
- Missing data techniques (missing data patterns to apply to the datasets);
- Datasets to run the models onto;
- Look-back window sizes;
- Forecasting horizons;
- Seeds to be used for the runs (one per configuration passed);
- CUDA devices (or the string "cpu") to run the pipelines in parallel on multiple devices.

As an example regarding the Cartesian product of parameters, given three models and two datasets, a total of six runs will be performed: each model on each dataset; the same applies to all other parameters, except for the CUDA devices.

By default, given the list of devices to run on, a process pool is created having two workers per device. Each process has its device assigned and cannot change it for the duration of the run. Having two processes per device allows one to start preprocessing a dataset while the other one runs the training, pipelining the training. The training phase is limited to one per device. If a process is already running in the training phase, the other process will wait for it to end. However, when training a model for imputation, only one model can be trained at the same time across imputation phases *and* training phases. As the imputation phase requires running a grid search for each new configuration, to speed up the grid search process all available devices are used to distribute the configurations to train. Once the best configuration is found (the grid search combinations are exhausted), an instance of the model using the best configuration is trained and the forecasting training phases are resumed. Below, a step-by-step description of the pipeline is given; this description corresponds to what a process spawned by the main pool executes, given an input configuration.

Step 1: Dataset Ingestion, Multi-source Detection and Alignment. The initial phase involves the instantiation of domain-specific dataset classes that extend a unified `BenchmarkDataset` base class. For each dataset, a class describing the procedure to load it needs to be provided; e.g., for the USHCN dataset there is a `USHCN` class that "knows" where to find the raw dataset and what operations are needed to prepare it for the main preprocessing phase. Usually, the first operation is loading the data in memory using pandas. Following, a core innovation of this framework: automated detection and handling of multi-source datasets. An algorithm checks whether there is an "ID" column and if there are multiple overlapping time intervals in the index. A dataset is categorized as multisource if it contains an "ID" column and exhibits overlapping temporal intervals across samples exceeding a 80% threshold. For such datasets (e.g., USHCN), each sample is treated as an independent entity and subjected to per-sample reindexing. This process ensures that disparate observation streams are aligned to a unified global temporal range, addressing the computational complexity inherent in non-aligned multivariate telemetry. Every dataset inherits common attributes by extending a main class representing a dataset, called `BenchmarkDataset`. This class contains all the common methods that are called for the preprocessing phase, stores information on the dataset itself, and intermediate versions of the dataset in memory during the processing.

Once the dataset instance is created, the main preprocessing pipeline is executed. The first step of the main preprocessing pipeline is the train-val-test split, respectively done using 70% of the data as the training set, 10% as the validation set, and 20% as the test set. This is done following the exact procedure taken from the implementation of models like DLinear and PatchTST [Nie+23b, Zen+22], to make the comparison even more accurate by replicating the exact data splits the authors used. If the dataset is multi-source, the split needs to be done for each sample, creating three DataFrames in total as the split is done on the temporal axis: the first 70% of the timestamps are taken for the training, the following 10% as validation, and the remainder is taken for the test set.

Each operation on the data is sanity-checked to make sure that even if there could be some error due to machine precision (handling floating data can introduce error in a variety of magnitudes depending on the domain), it keeps under a certain threshold. The execution is interrupted if one of the operations is not performed correctly; this is noted by comparing the

output of a function with the same operation applied to data that is known to be correct; if the two arrays or DataFrames match (i.e., the error is smaller than a very small ε like 10^{-9}) then the original data is overwritten with the modified version, so to contain memory usage. This is particularly important for multi-source datasets, where the same operation has to be performed on the dataset split on samples: if, for any reason, the data is not correctly aligned, the samples will have an error that can be highlighted to the user to investigate.

Step 2: Stochastic Corruption (The "Dirtying" Phase). Using a set of fixed random seeds, the framework removes data points from the dataset, which is also the ground truth used for the training. This creates a controlled "irregular" environment where the percentage of missingness is known, but the pattern varies per run to test model sensitivity. Implemented missingness patterns are:

- Random (single-point "hole")
- Per timestamp (all features on a given timestamp are missing)
- Sequence (contiguous sequences of specified length are randomly deleted from the features columns until the specified missingness percentage is reached)
- Block (a percentage of the data is deleted in square or rectangular shapes, creating large "holes" of missing data)

For the *sequence* and *block* missingness patterns, the size of the sequence, as well as the block length and block width, are computed as $\max(1, \#\text{columns} - 2)$. The implementation for these two patterns is taken from the **PyGrinder** library, a part of the **PyPOTS** ecosystem, focused on creating missingness in Time Series data [Du+24, Du+25]. Since PyGrinder, as the whole PyPOTS ecosystem, uses data in the shape of **non-overlapping sliding windows**, it requires the data to be transformed ad hoc with an additional step. Data is then transformed back to a tabular shape once the missingness has been created.

Step 3: Scaling. Once the dataset has been "dirtied", it can be standardized or normalized. This is done only at this point in the pipeline because this is the version that the imputation or forecasting models will actually use. If the scaling were performed before the missingness addition, the data would be scaled considering points that the models don't actually have available, thus inducing some bias in the models' learning process, erroneously improving their performance.

Step 4: Imputation and Grid Search. At this point, the dataset has been loaded, split, dirtied, and scaled, so all the necessary steps to recreate a specific scenario have been executed: the dataset from now on is a specific version where data is missing in a specific percentage with a specified pattern, and it is scaled accordingly. The next step is, if the forecasting model requires it, to fill the holes back again. Assuming this is the kind of dataset at hand, the only thing remaining is to try and "guess", depending on which filling technique was chosen, what data was present in the holes.

The imputation techniques are divided into *simple* and *complex*. Simple techniques are very fast, but usually less accurate:

- Linear
- Mean
- Median
- LOCF (Last Observation Carried Forward)

Complex techniques are DL models specifically designed for imputation, or otherwise adapted for this downstream task:

- SAITS [DCL23]
- BRITS [Cao+18]
- TimesNet [Wu+23]
- StemGNN [Cao+21]
- DLinear [Zen+22]
- FreTS [You+25]
- CSDI [Tas+21]
- USGAN [AK23]

Complex techniques' architectures were chosen to have a variety of mechanisms to compare across different domains and missingness patterns. All implementations are taken from **PyPOTS**'s imputation module [Du+25].

If a complex imputation technique is selected, the framework executes a **grid search** on the model's hyperparameters and look-back window sizes. The hyperparameter configurations were taken from PyPOTS' paper, where the authors show how they tuned the models for each dataset [Du+24, Du+25]. As this framework already included BeijingPM2.5 (in its single-source version), Electricity, and ETTh1, all possible configurations are enumerated and tested across the other datasets as well. The look-back windows sizes were inspired by DLinear's paper [Zen+22]:

- Hourly or minutely sampled (e.g., ETTh1): 96, 192, 384, 768
- Daily sampled (e.g., ExchangeRate): 96, 192, 360. 720
- Small datasets (e.g., ILI): 24, 36, 48, 60

These values for the look-back window sizes enable the pipeline to handle all types of datasets at hand, adapting to the sampling rate and dataset size. Each hyperparameter configuration is tested on each look-back window size.

The framework adopts a sophisticated procedure to decide how many configurations can be run on the given devices and how to distribute them. First of all, as previously mentioned, all processes needing GPUs are obliged to wait on a specific lock for the imputation procedure; this mutually excludes training forecasting models when an imputation process has to be carried on.

When the system has been "isolated" from the other processes, the process responsible for the currently active imputation procedure does the following for each available device:

- Create an instance of the model to train
- Move the model and a dummy tensor of shape `(batch size, num_past, num_features)` (to emulate the amount of memory that will be necessary for a batch of data) to the device
- Run a training epoch
- Compute the amount of VRAM used

Based on the amount of consumed VRAM, the framework decides how many configurations to run on each device by dividing the total available VRAM by the consumption; to avoid computational and I/O transfer bottlenecks, at most 4 workers are assigned to each device.

At this point, a pool of processes is created for each device. For each configuration computed for the grid search, a task is created and assigned in a round-robin fashion to the pools to distribute the workload.

As some specific configurations of hyperparameters have been observed to be particularly unstable regarding VRAM usage, for each epoch a CPU fallback was implemented. This way, the training can continue at a slower pace, but the progress is not lost. After each epoch, the worker tries to go back to the GPU, in case some other process exited and freed some VRAM.

To evaluate all imputation techniques, a "dirtier" version of the dataset is generated, having 10% more missing data. The dirtier validation set is necessary to evaluate the inference error of an imputation model: to compute an error metric, it is mandatory to know which data was previously present in the dataset as ground truth. This way, at each epoch, the imputation model is evaluated considering the MSE computed on the whole validation set, with a mask that allows the system to consider only the specific 10% of additional missing data. The chosen percentage of 10% more missing data for the validation set was estimated considering that the most tested missingness configurations were having 20%, 50% and 80% missing data. Therefore, in the worst case, the validation set would have 90% missing data. For the simple imputation methods, the error is computed just once, as there is no training procedure to be done.

This ensures that the forecasting model's performance isn't throttled by a poorly tuned imputation step. The best hyperparameter configuration (i.e., the one yielding the lowest MSE loss on the validation set) is then trained, and the dirtied data is finally imputed and subsequently used for the sliding windows.

This meticulous approach to the imputation layer serves a critical purpose in the benchmarking philosophy. By providing the "standard" models (like LSTM, TCN, or PatchTST) with the highest-quality imputed data possible, a "level playing field" is created. This enables the observation of whether native ISTS models (like GRU-D or mTAN) truly offer an architectural advantage by modeling Δt directly, or if a well-tuned "Standard + Imputed" pipeline can remain competitive across heterogeneous domains.

Step 5: Shuffled Window Training. Once the data has been imputed, it is converted back from the non-overlapping sliding window format to the tabular DataFrame format. For the correctness of the forecasting training phase, the dataset labels with missing data are kept on the side, as the imputed values of the target series are useful for the past, but cannot be used to evaluate the forecasting accuracy in the future. The data is then transformed into a set of **overlapping sliding windows** with an offset of 1 time point.

Because of the computational complexity of creating many sliding windows, especially for a multi-source dataset, the algorithm has been parallelized to exploit as much as possible of the available CPU cores. Each core works on a group of samples, and the resulting windows are subsequently concatenated based on their sample ID.

As for the datasets, each model has its own class, which can be customized in case the sliding window dataset needs to be transformed further because of specific model requirements. Each model extends the `BenchmarkModel` class, where common attributes and information is kept, like the reference to the dataset instance.

To mitigate the Fixed Origin Fallacy, the training set's windows are shuffled before being fed to the model, preventing the architecture from overfitting to a specific chronological sequence and forcing it to learn universal temporal dynamics.

Step 6: Terminal Point Evaluation. The forecasting phase is divided into two functions: `train` and `test`. During the `test`, once the training is completed, the model generates a forecast for the horizon H for each window in the test set. The framework extracts only the value at $H_{\{\max\}}$. This process is repeated across the whole test set to generate the final robust performance statistics by comparing the array of forecast values with the ground truth for each time step.

To ensure the scientific integrity of the results, the framework enforces a strict anti-leakage protocol:

- The train-validation-test split (70/10/20) is performed chronologically before any scaling or imputation occurs.
- Standardization and normalization parameters (mean, variance) are calculated strictly on the training set and then applied to the validation and test sets. This prevents "future information" from leaking into the training phase through global statistics.
- During the forecasting phase, while imputed values are used for past context, the terminal point evaluation is performed strictly against the original, non-imputed ground truth labels to ensure metrics reflect real-world predictive power.

By combining multisource alignment, stochastic sensitivity testing, and optimized decoupled imputation, this framework provides a rigorous environment for challenging the current state-of-the-art. As a consequence of the No Free Lunch Theorem, we can expect some configurations to work better than others on specific domains; in fact, this is one of the research questions the benchmark aims to answer.

3.6.2 Experimental Rigor

To ensure the scientific integrity of this benchmark and to facilitate complete experimental reproducibility, the framework adopts a multi-layered protocol designed to eliminate common pitfalls in TSA, such as data leakage and stochastic bias. By strictly decoupling the scaling parameters from the test environment and enforcing chronological data partitioning, we ensure that the performance metrics reported in the subsequent sections reflect genuine predictive generalization rather than statistical artifacts. The following Table 3.4 summarizes the safeguards enforced across all experimental runs of this campaign.

Criterion	Safeguard Mechanism
Data Leakage	Chronological 70/10/20 splitting; normalization/standardization parameters derived strictly from the training set.
Ground Truth Integrity	Final evaluation is performed exclusively against the original, non-imputed ground truth labels at the H_{max} terminal point.
Stochastic Robustness	All experiments are executed across five fixed random seeds (0-4) to isolate performance from specific missingness realizations.
Hyperparameter Policy	Use of author-recommended settings for baselines or automated grid searches for heterogeneous datasets.
Statistical Validation	Use of non-parametric Friedman tests ($p < 0.0001$) and Nemenyi post-hoc analysis to validate performance hierarchies.
Hardware & Stack	Standardized execution environment using AMD EPYC 7702 CPUs, NVIDIA A100 GPUs and a documented C++/Python stack for temporal consistency.

Table 3.4: Reproducibility and Validity Checklist to summarize the action taken towards accurate scientific results.

3.7 Results and Discussion

All metrics reported in this section are calculated in the standardized feature space. This methodological choice ensures that performance comparisons are not biased by the differing scales of heterogeneous domains — such as the high-magnitude energy values in Electricity versus the low-magnitude coefficients in ExchangeRate — thereby isolating the architectural efficacy of the models.

3.7.1 Research Question 1: ISTS models or Imputation + Standard Forecasting?

The first question that this benchmark wants to answer is how to effectively process irregularly-sampled datasets. Literature offers many ways to proceed, but in practice, what yields the best results? What performs better on a specific domain? Do models specifically engineered for irregular sampling (e.g., GRU-D, mTAN) offer a tangible architectural advantage over standard forecasting architectures (e.g., PatchTST, DLinear) that rely on an external imputation layer?

To answer these questions, the first part of the experiments had the following objective: **compare standard forecasting using a simple imputation method with Native ISTS forecasting**. For *RQ1.1* a total of 2880 experiments were run with the following parameters:

- NaN mechanism: random
- NaN percentages: 20%, 50%, 80%
- Look-back window size: 96
- Forecasting horizons: 96, 192, 336, 720
- Imputation technique: LOCF
- Datasets: USHCN, FrenchPiezo, BeijingPM2.5, ETTh1, ILI, ExchangeRate
- Models: PrimeNet, GRU-D, mTAN, NHiTS, DeepAR, TCN, DLinear, PatchTST
- Seeds: 0 to 4

The datasets were chosen to have three multi-source and three single-source. The models were chosen with the following logic:

- PrimeNet, GRU-D, and mTAN are three **Native ISTS** models
- All the others are standard models, but with different architectures:
  - *Multi Layer Perceptron*: NHiTS, DLinear
  - *Recurrent Neural Network*: DeepAR
  - *Convolutional Neural Network*: TCN
  - *Transformer*: PatchTST

The stochastic perturbation technique was fixed to random, as well as the look-back window size was fixed to 96, because these parameters are not relevant to answer the main question, so they were removed from the variables contributing to the degrees of freedom.

To visualize the comparison between Native ISTS models and Imputation + Standard pipelines, the average metric performance is plotted as grouped bars across the forecasting horizons for a fixed missing data percentage in Figure 3.1a Figure 3.1a Figure 3.1a. MAE was chosen as it does not magnify error like MSE.

The comparative analysis between the families of models reveals a consistent architectural advantage for specialized models, except for the scenario with 80% missing data. However, the magnitude of this advantage is heavily influenced by specific dataset characteristics. As illustrated in the global family plots (Figure 3.1), the Native ISTS family (orange bars) consistently achieves lower MAE across all evaluated horizons where the missing data percentage is 20% and 50%. While Native ISTS models' performance decreased drastically for the 80% missing data scenario, Imputation + Standard models pipelines appear to be more resilient. The error doesn't vary much across the forecasting horizons, but looking at Figure 3.2, it is noticeable that the R^2 Score for Native ISTS models is way higher — or less negative — than Imputation + Standard models pipelines, showing that even if the difference gap in MAE metric is not as big as one would expect, the forecasting from Native ISTS models is more trustworthy. The same trend of Figure 3.1 persists in Figure 3.2, suggesting that the explicit modeling of Δt and relational propagation (as seen in GRU-D and PrimeNet) loses its efficacy when there is so little data carrying information; imputed data, even if it carries along some error, is more effective for the learning process for long-term forecasting than forecasting on unknown data.

To have an overview of the models' performance across all datasets, Figure 3.5e reports the performance in a radar fashion, allowing for an easier comparison. The most significant finding regarding Research Question 1.1 was thus discovered: the impact of ILI and ExchangeRate datasets on global metrics. In Figure 3.2, the presence of these two datasets causes the average R^2 scores to plummet into negative territory. This highlights a "failure regime" where the complexity of the domain outweighs the models' current learning capacity. When ILI and ExchangeRate are excluded from the statistics, the architectural "landscape" shifts dramatically.

As shown in Figure 3.4, the R^2 scores for both families transition into the positive range, indicating genuine predictive generalization, except for the 80% missing data percentage scenario, where Native ISTS models have so little data to learn from. In this "cleaned" distribution, Native ISTS models transition into positive R^2 ranges, yielding higher scores than the standard pipelines. This mirrors the behavior seen in the high-noise regimes, where they were "less negative," reinforcing the conclusion that specialized architectures better recover the underlying temporal signal. Figure 3.3c, Figure 3.3c, and Figure 3.3c also show that the gap between Imputation + Standard pipelines and Native ISTS models is smaller, indicating that the difficulty of ILI and ExchangeRate lowered the performance of both families, with a higher impact on Imputation + Standard models pipelines; nonetheless, the general behavior observed among the two families is maintained. While the "Standard + Imputation" pipelines are highly competitive in infrastructure-heavy domains like ETT, they are more susceptible to the artifacts introduced during the imputation phase. The Native ISTS models, by processing raw irregular data, avoid this propagation of error. However, the more data is missing, the more Native ISTS models struggle to catch the underlying temporal dynamics. The narrowing of the metric gap when removing "hard" datasets suggests that for many standard multivariate tasks, the performance gain of native models may be marginal compared to the speed and ease of use provided by high-performance standard models paired with simple imputation. Native ISTS models aren't

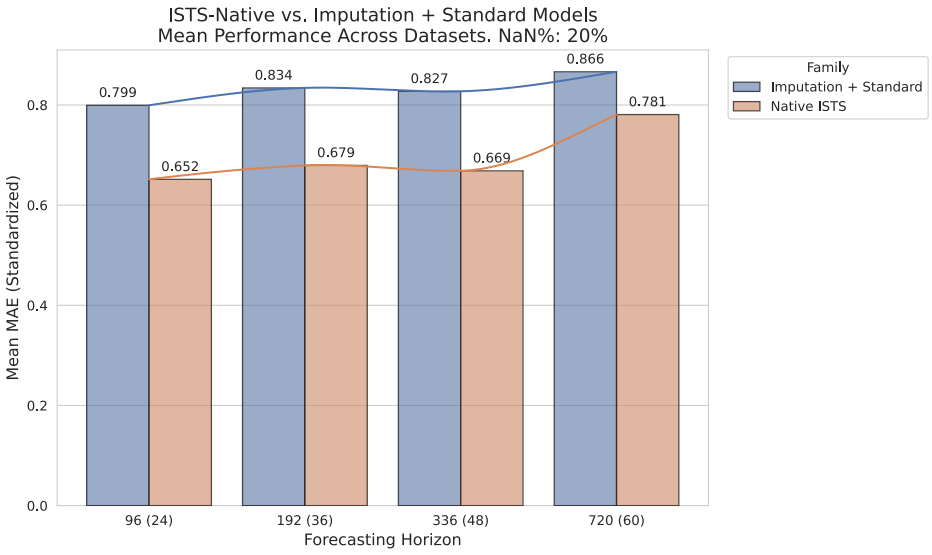
just "more accurate" – they are more dependable. They offer generally better R^2, justifying a potentially higher computational expense when data is not extremely scarce.

An in-depth analysis of the plots can be read in Section A.3.1.

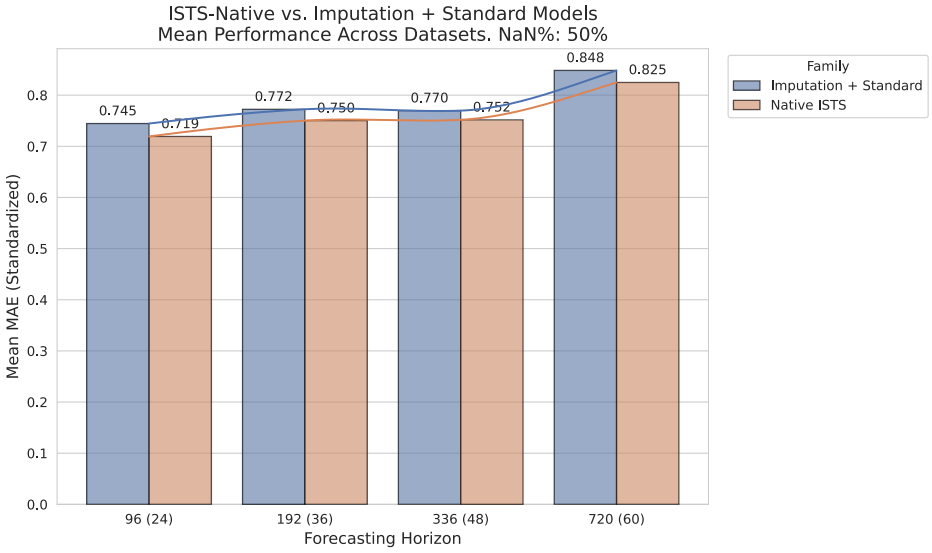
3.7.1.1 "No Free Lunch" Observations: Architectural Failures and Successes

Consistent with the No Free Lunch theorem, no single architecture dominated all eleven datasets. Figure 3.5e shows it as an overview across all datasets, missing data percentages, and forecasting horizons. All models showed similar behavior metric-wise per dataset, indicating no global winner. However, the reliability of the model's performance is better compared through a range plot of the R^2 Score in Figure 3.5e. The plot shows that every model had similar forecasting significance behavior per dataset (i.e., all models struggled with forecasting reliability on BeijingPM2.5 and ETTh1, while having an easier learning process on USHCN and French-Piezo); nonetheless, it is noticeable how all models also struggled on ILI and ExchangeRate, where the R^2 Score is always negative. Some models, like GRU-D, struggled more than others on all datasets (on average), whereas PrimeNet and TCN are the ones that generated more trustworthy predictions across all scenarios. Surprisingly, PrimeNet's architecture didn't suffer much in performance across the 80% missing data scenario, even though it is a Native ISTS model. This demonstrates greater architectural resilience compared to GRU-D, where the dataset complexity significantly influenced its performance in producing reliable predictions.

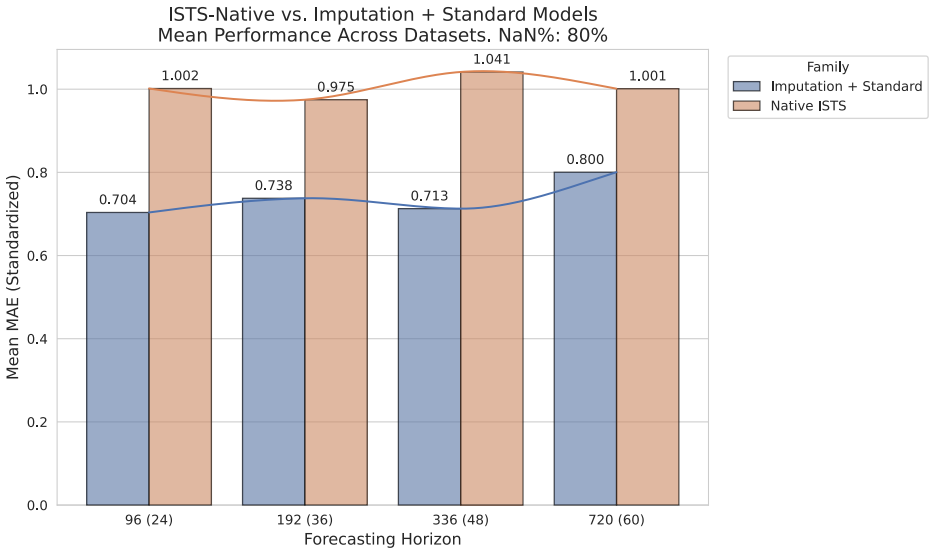
An in-depth report of radar plots regarding the relationship between models' performance metrics and reliability can be found in Section A.3.2.



(a)

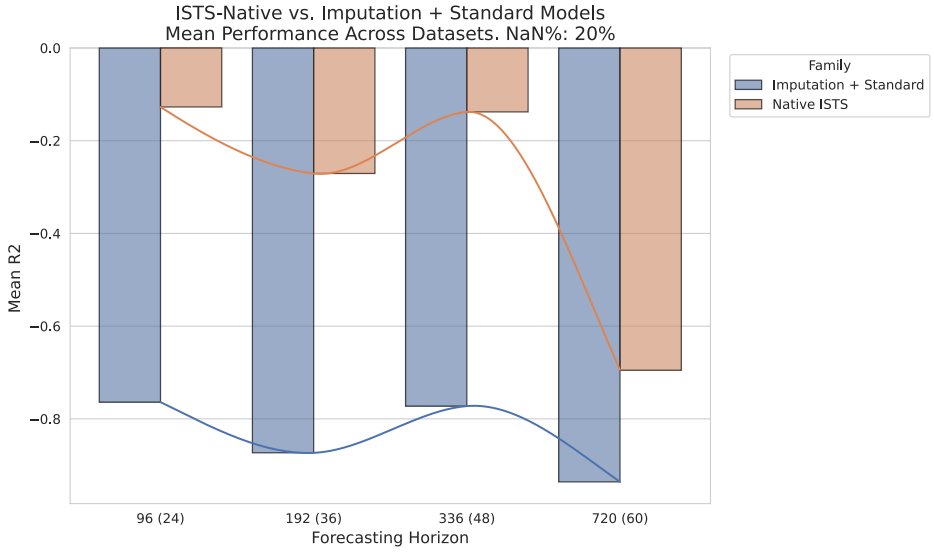


(b)

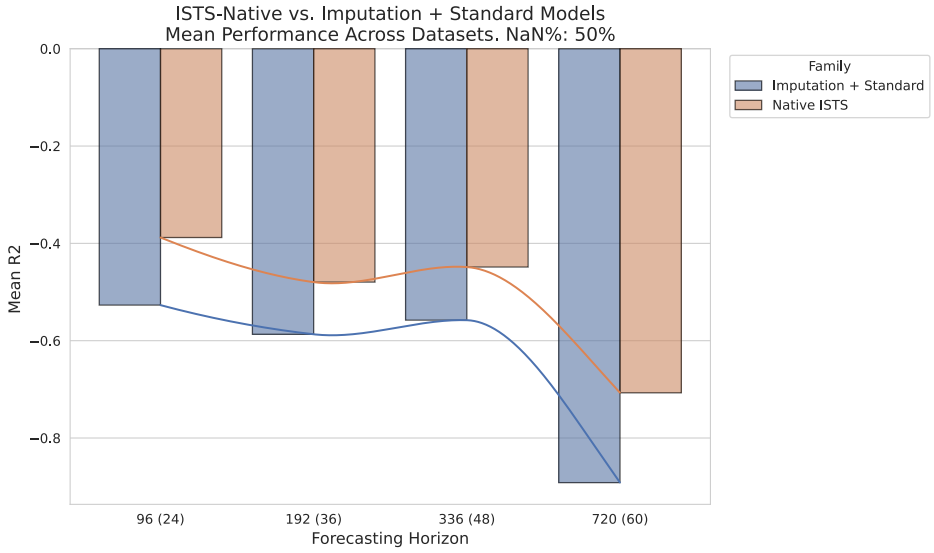


(c)

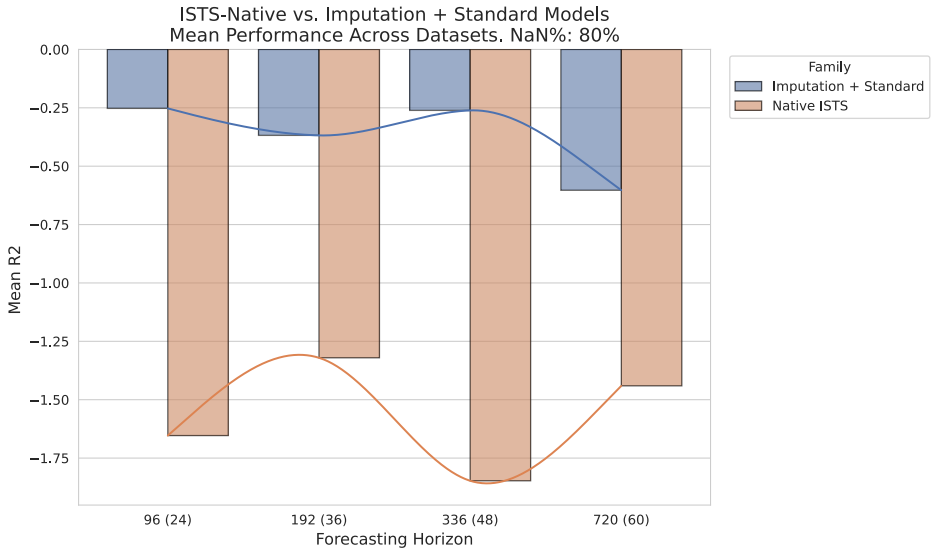
Figure 3.1: Native ISTS models MAE performance comparison with Imputation + Standard Forecasting Models for 20%, 50% and 80% missing data percentages across forecasting horizons.



(a)

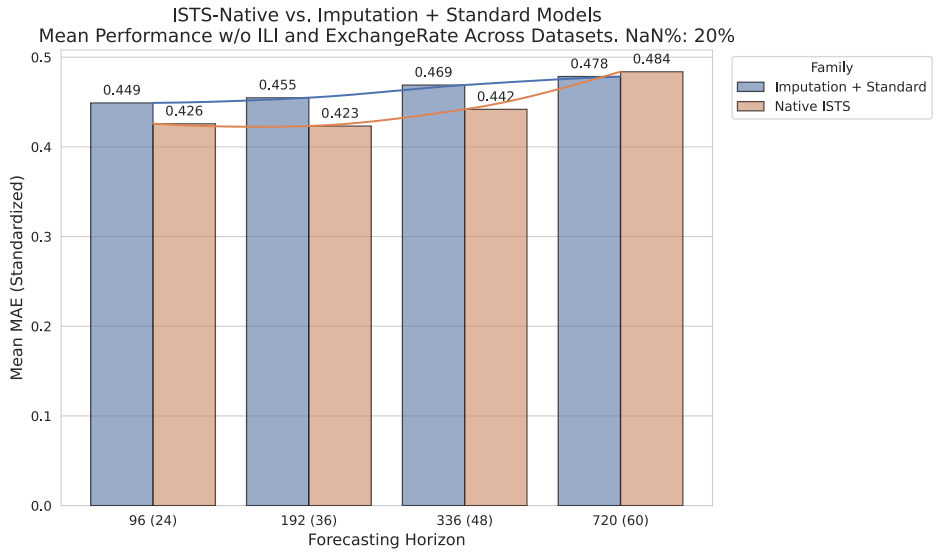


(b)

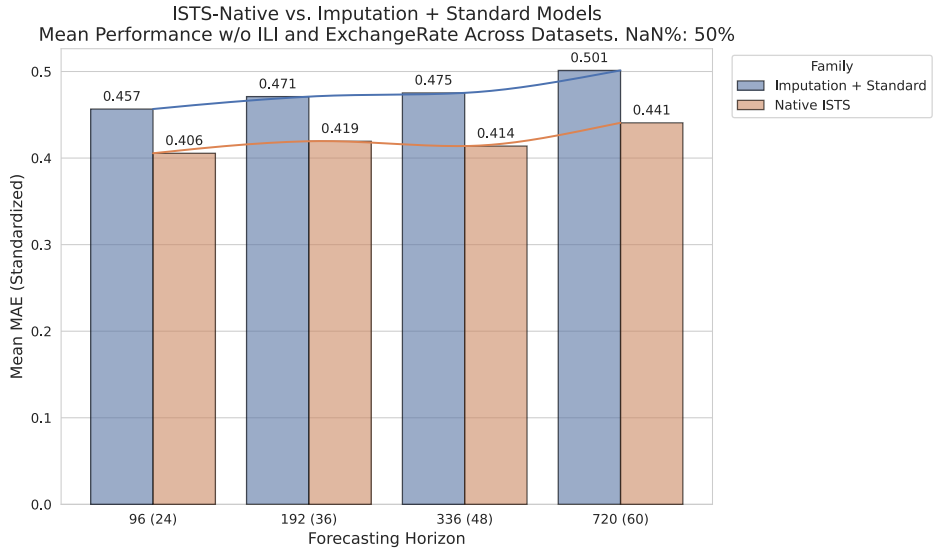


(c)

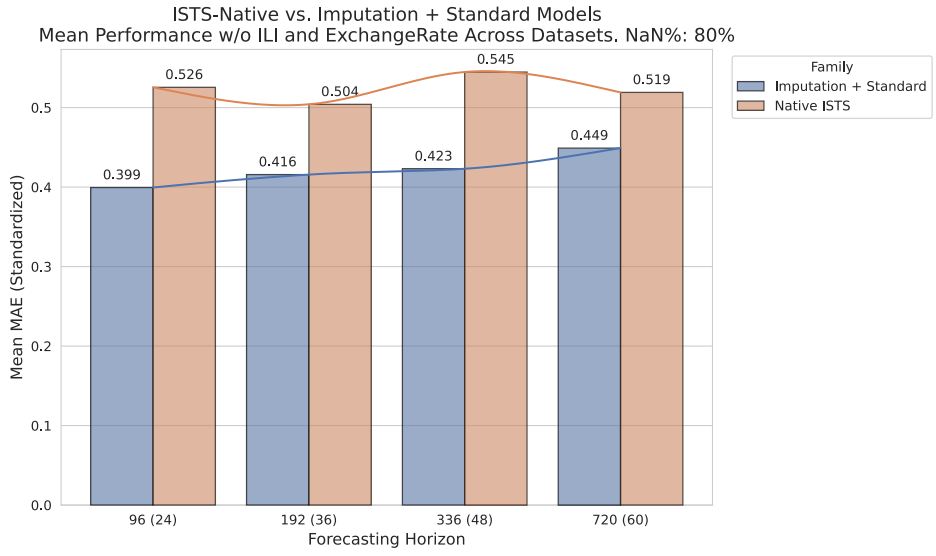
Figure 3.2: Native ISTS models R^2 Score compared to Imputation + Standard Forecasting Models for 20%, 50% and 80% missing data percentages across forecasting horizons.



(a)

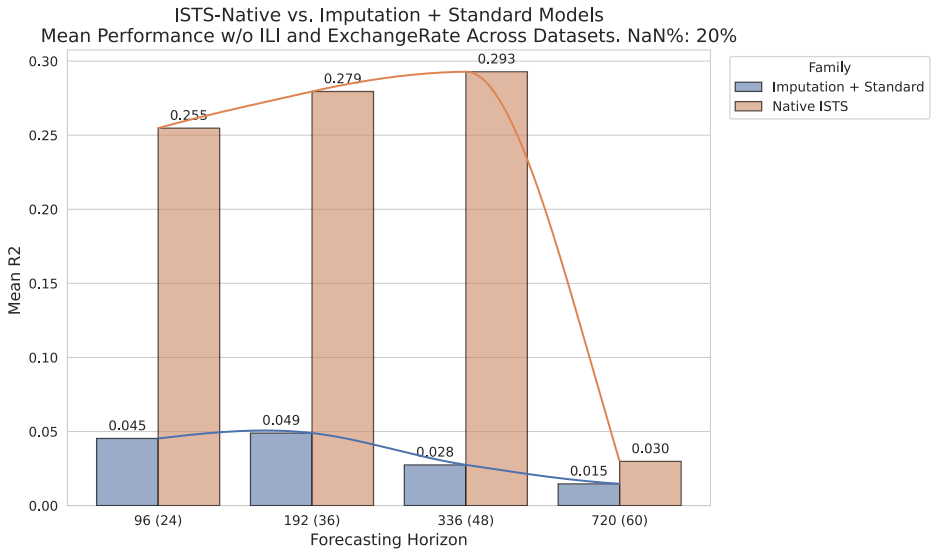


(b)

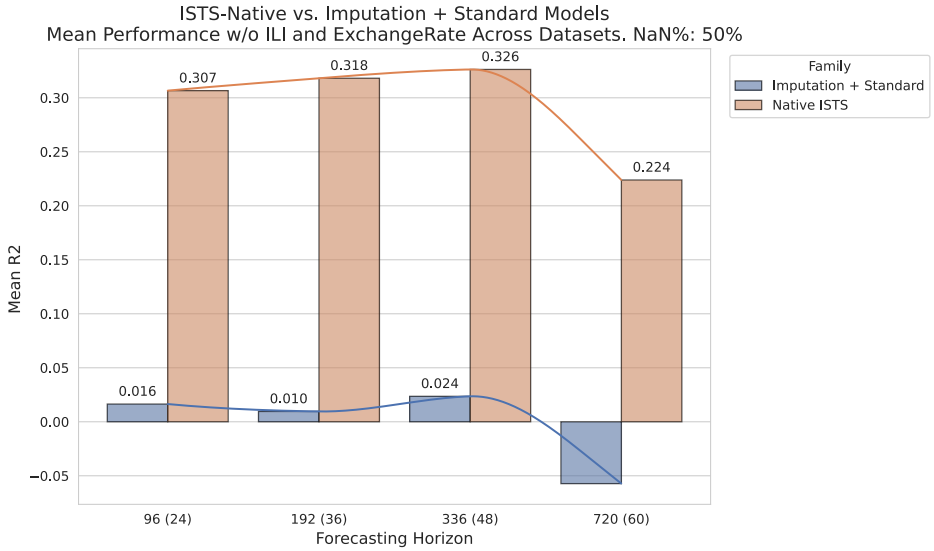


(c)

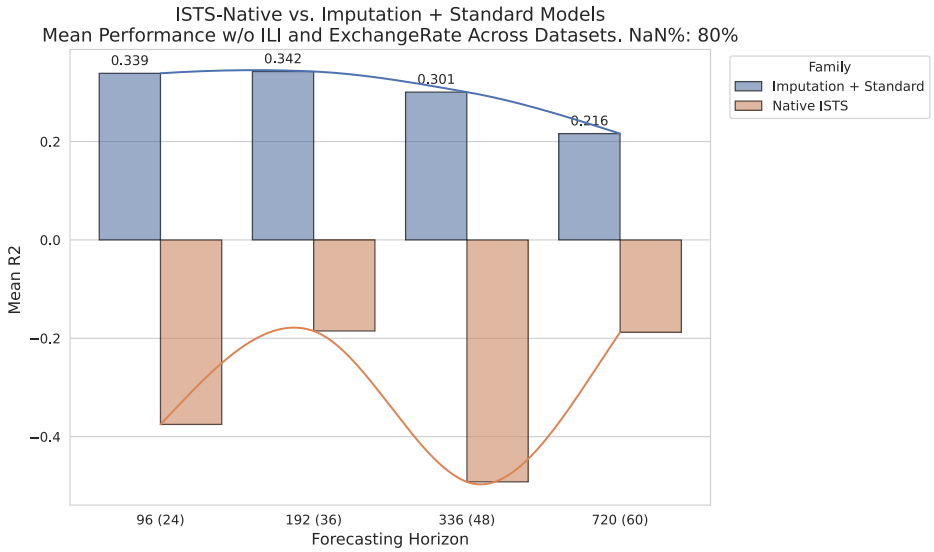
Figure 3.3: Native ISTS models R^2 Score compared to Imputation + Standard Forecasting Models for 20%, 50% and 80% missing data percentages across forecasting horizons **without** ILI and ExchangeRate datasets.



(a)

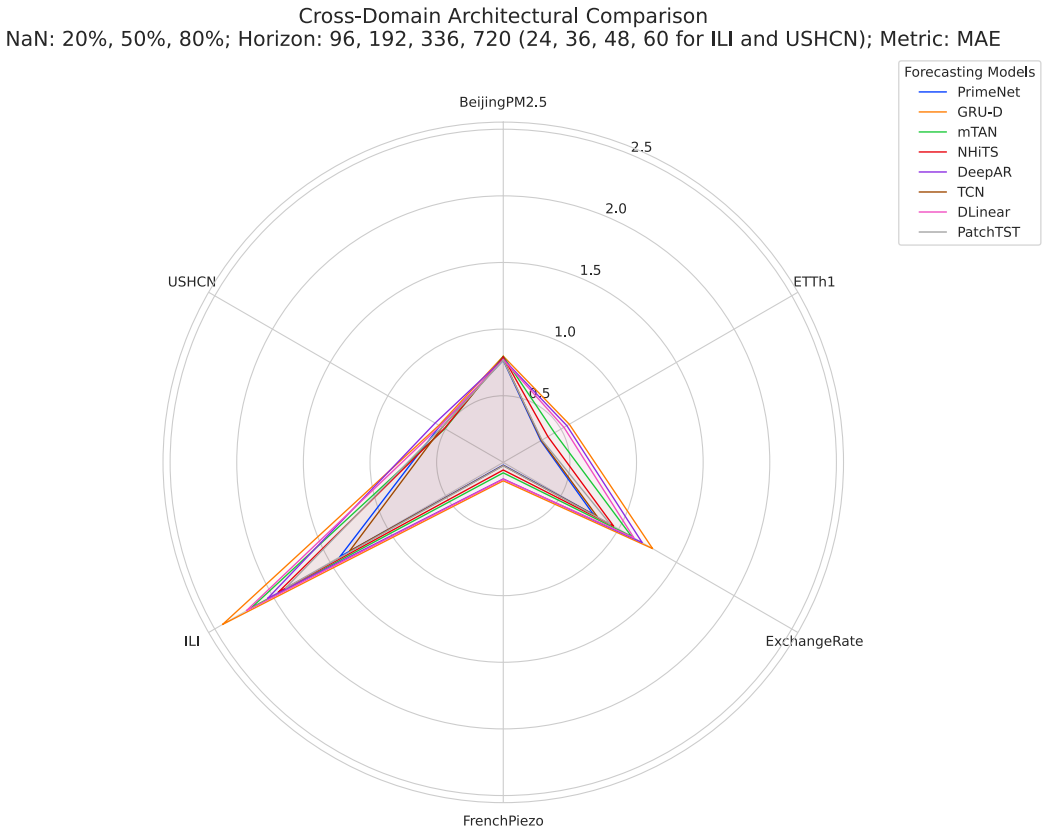


(b)

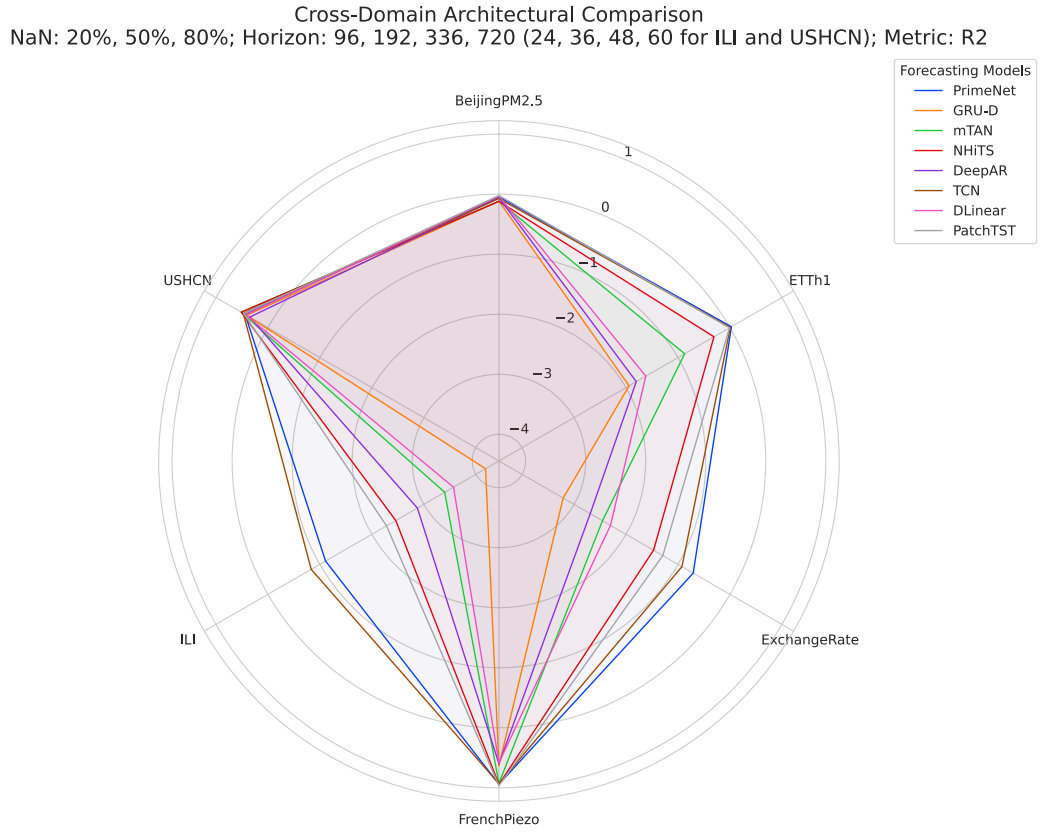


(c)

Figure 3.4: Native ISTS models R^2 Score compared to Imputation + Standard Forecasting Models for 20%, 50% and 80% missing data percentages across forecasting horizons **without** ILI and ExchangeRate datasets.



(a)



(b)

Figure 3.5: Radar plot visualizing the average MAE (a) and R^2 (b) performance of models over each dataset.

3.7.1.2 Statistical Validation Across ISTS Scenarios

To ensure the scientific validity of the architectural comparisons, Critical Difference (CD) diagrams based on the Nemenyi post-hoc test are employed. While raw metric averages (MAE/ MSE) provide an initial performance ranking, they do not account for the variance across heterogeneous datasets. The CD diagram addresses this by identifying groups of models whose performance is not statistically distinguishable at a significance level of $\alpha = 0.05$. This allows for an objective determination of whether the perceived superiority of specific architectures is a robust phenomenon or a domain-specific artifact.Figure 3.6 reports the MAE metric for all models across all scenarios. The plot identifies two primary cliques: (i) PrimeNet, TCN, and PatchTST; and (ii) DLinear, NHITS, DeepAR, mTAN, and GRU-D. This hierarchy is concordant with the global performance summarized in Figure 3.5e: PrimeNet, TCN, and PatchTST demonstrate greater adaptability across various domains, consistently yielding higher R^2 scores and lower MAE. Conversely, GRU-D, mTAN, and DeepAR fall into lower-performing ranks on average. This suggests that in a 'one-size-fits-all' global evaluation, the complex temporal modeling of Native ISTS architectures may not always provide a statistically significant edge over highly optimized standard pipelines.

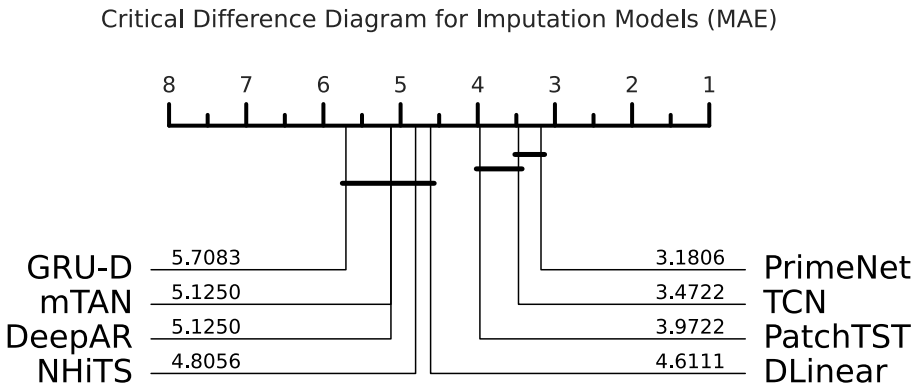
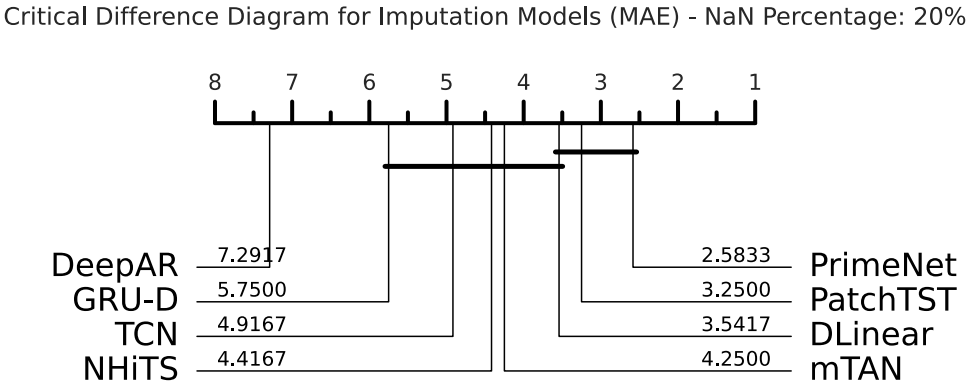


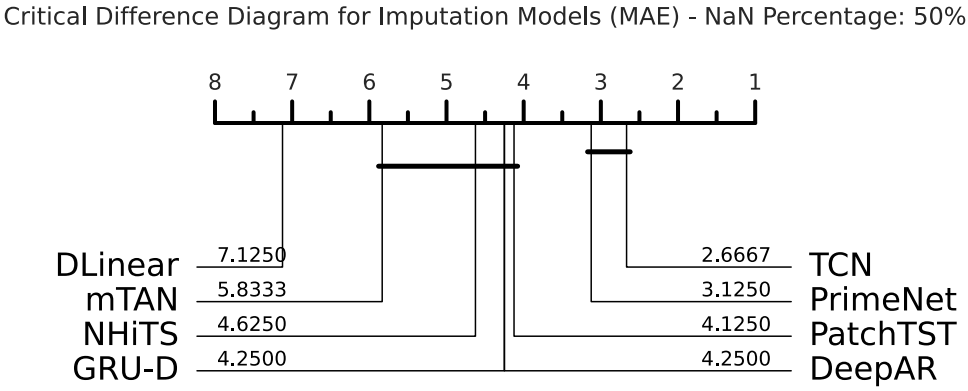
Figure 3.6: Critical Difference diagram across all missing data percentages and forecasting horizons.

To further analyze missing data dynamics, Figure 3.7 illustrates the CD diagrams for the 20%, 50%, and 80% sparsity scenarios. At 20% missingness, PrimeNet and PatchTST dominate, reinforcing the observation that standard architectures paired with optimized imputation are highly effective when the temporal signal remains largely intact. In medium sparsity scenarios (50%), TCN transitions to the top rank, suggesting that its local, dilated convolutional inductive bias is more robust to mid-level corruption than the global attention mechanisms of Transformers.

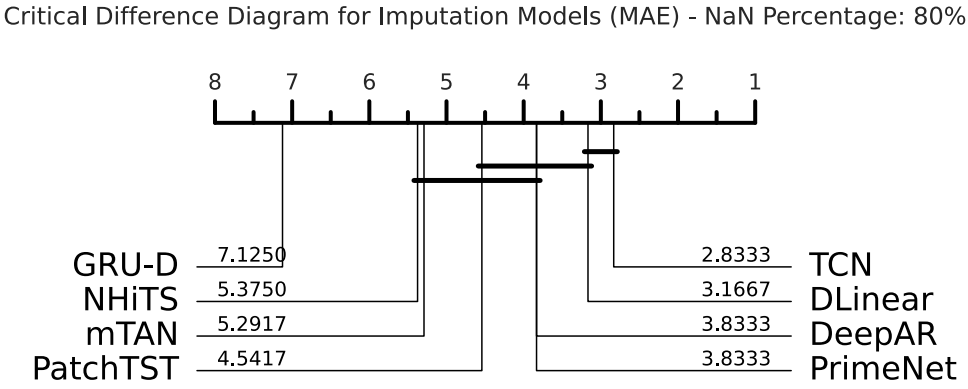
In high sparsity scenarios (80%), the hierarchy shifts significantly again. TCN and DLinear occupy the leading ranks; it is particularly noteworthy that DLinear — a simple linear mapping — becomes statistically competitive under extreme sparsity. This reinforces the conclusion that complex attention mechanisms can become unstable when the majority of the input sequence is estimated by the imputation layer, introducing too much error, or when it is missing completely.



(a)



(b)



(c)

Figure 3.7: Critical Difference diagrams reporting the statistical relevance of models across 20%, 50%, and 80% missing data scenarios.

The statistical rankings provided by the CD diagrams find a direct correspondence in the R^2 Reliability Quadrants plots. A general plot considering all missing data percentages and forecasting horizon hasn't been reported, as it was considered insignificant: the data showed a

behavior that does not represent the actual classification for the models in the more fine-grained versions of the plot. Therefore, in Section A.3.5 all plots are reported for correctness.

While the CD analysis across all scenarios (Figure 3.6) indicates that PrimeNet, TCN, and PatchTST occupy the top-performing clique, the quadrant distribution reveals a deeper qualitative divide in architectural resilience. Across the majority of scenarios, such as the 80% NaN regime (Figure A.39 Figure A.40), PrimeNet and PatchTST consistently populate the Target Zone (Quadrant II), maintaining positive explanatory power even as MAE increases. Conversely, models in the lower-performing CD ranks, such as GRU-D and mTAN, frequently collapse into Quadrant IV (Failure), where R^2 values drop significantly below zero, reaching as low as -7.0 in high-sparsity configurations (Figure A.39 Figure A.40). This demonstrates that the statistical superiority of top-ranked models is rooted in their ability to maintain a predictive "floor" where others experience total learning failure.

3.7.1.3 RQ1.1 Summary

The results of this benchmark confirm that there is no universal answer to whether Native ISTS models or Imputation + Standard pipelines are superior, validating the No Free Lunch theorem across irregularly sampled data regimes. No single architecture dominated all six datasets simultaneously, as every model demonstrated specific strengths and weaknesses contingent upon the data domain, sparsity level, and forecasting horizon.

Summary of Key Findings.

1. While Imputation + Standard pipelines often prove highly competitive in infrastructure-heavy domains or low-sparsity scenarios (20% NaN), they are more susceptible to error propagation introduced during the imputation phase. In contrast, Native ISTS models are often more dependable and stable, maintaining a predictive "floor" and preventing extreme outliers in high-variance or sparse domains.
2. Native ISTS models (e.g., PrimeNet, GRU-D, mTAN) exhibit a consistent architectural advantage at 20% and 50% missingness by achieving lower MAE and higher R^2 scores. However, their efficacy decreases drastically at 80% missing data, where standard pipelines often show greater resilience.
3. Even when the gap in MAE is narrow, Native ISTS models frequently yield significantly higher — or less negative — R^2 scores, making their forecasts more trustworthy than those of standard pipelines. In "cleaned" distributions (excluding skewing datasets like ILI and ExchangeRate), Native ISTS models transition into positive R^2 ranges, indicating genuine predictive generalization.
4. Statistical Significance: CD diagrams identify PrimeNet, TCN, and PatchTST as the top-performing clique, demonstrating the greatest adaptability across diverse domains. As sparsity reaches 80%, simpler architectures like DLinear become statistically competitive, suggesting that complex attention mechanisms can become unstable when input sequences are largely estimated or missing.

The Verdict. The choice between these two approaches depends largely on the specific operational requirements:

- Choose Imputation + Standard Forecasting for standard multivariate tasks or infrastructure domains where computational efficiency, speed, and ease of use are prioritized. Using a fast imputation technique like LOCF with a standard model can provide accurate results with significantly lower training times and costs
- Choose Native ISTS Models when dealing with complex, sparse, or high-variance domains where reliability is critical. Despite potentially higher computational expense, the ability of these models to recover the underlying temporal signal and maintain architectural resilience justifies their use when data is not extremely scarce.

3.7.2 Research Question 2: Do complex imputation methods improve forecasting performance?

The objective of RQ2 is to evaluate whether transitioning from simple, low-cost imputation (e.g., LOCF) to more complex, deep-learning-based imputation methods (e.g., SAITS, BRITS [Cao+18, DCL23]) significantly enhances the downstream performance of standard forecasting models. Similar to the "cliques" found in RQ1.1, we categorize imputation methods into families (Simple Methods and Complex Methods).

As for RQ1, the research was split into multiple subquestions. RQ2.1 focuses on answering **which are the top 3 imputation methods** across a selection of 10 approaches. The experiments' configuration is as follows, for a total of 600 runs:

- NaN mechanism: random
- NaN percentages: 20%, 50%, 80%
- Look-back window size: 48
- Forecasting horizon: 96
- Imputation techniques:
  - Simple Methods: LOCF, Mean, Linear.
  - Complex Methods: SAITS, BRITS, USGAN, TimesNet, DLinear, StemGNN, FreTS.
- Datasets: ETTh1, ExchangeRate, USHCN, ILI
- Model: DLinear
- Seeds: 0 to 4

The look-back window size and forecasting horizon were fixed to a value that allowed the sliding windows to be created on every chosen dataset; the values related to forecasting, as well as the model (DLinear), are fixed, as the objective of this research question is not to study the

downstream forecasting results, but the performance of imputation methods in terms of error (MAE / MSE) directly on removed data to determine which method provides the most faithful reconstruction of the original signal. DLinear was chosen as it is the most lightweight model among the available ones in the benchmark. Once the top 3 imputation methods are identified, the next step would be to select the one that improves imputation + standard model pipelines the most against LOCF, used as a baseline. The best complex method would then be used to compare the results of the forecasting task using what was gathered during RQ1.1, which used LOCF as an imputation method.

To be properly compared, the metrics are reported in the standardized space, exactly as for RQ1. All methods are taken from the PyPOTS library and the corresponding paper [Du+24, Du+25]. Therefore, since the authors ran multiple experiments to improve the models' performance on the dataset at hand, the same is done for this benchmark: in Section A.4.1, the best hyperparameters' configurations found by the authors are reported for every complex method, relatively to the three datasets that the authors used: BeijingPM2.5 (in the single-source version [Zha+17]), Electricity and ETTh1. For the sake of correctness, if the relative experiment configuration's best hyperparameters file is missing, a **grid search** process is launched to determine the best combination for different datasets as well. This is also done because a framework-supported grid search, as the one done by the authors of the PyPOTS library, where Microsoft's Neural Network Intelligence (NNI) framework was adopted [Mic21], would have been extremely time-consuming.

In RQ2.1, the analysis examines whether the Simple Methods family remains a "hard baseline" to beat across most datasets. Scenarios where complex methods actually degrade performance by introducing "synthetic noise" or "imputation artifacts" are highlighted, as these can confuse the forecasting model.

Using the same structure as the CD diagrams in Figure 3.7, the "best" imputation methods are examined to observe the shift caused by the missing data percentages:

- 20% NaN: Does complex imputation provide any meaningful gain when data is mostly present?
- 80% NaN: At extreme sparsity, does the "hallucination" of complex models become more or less reliable than simple constant-value carryover (LOCF) or similar approaches?

Figure 3.8 illustrates a comparison of the two imputation methods' families through grouped-bar plots, the former in a general performance context, and the latter across different missing data percentages. On a metric (MAE) level, complex imputation methods achieve lower MAE than simple imputation methods. On the other hand, the total time taken by the run (considering the imputation model training and inference time) should be taken into consideration to evaluate the effective usefulness of the method. Moreover, as a result of Section 3.7.1, using DL imputation methods might be overkill for 20% and 50% missing data scenarios. Complex imputation methods are most useful in scenarios with 80% missing data, where ISTS Native models struggle to capture the underlying data dynamics due to their scarcity.

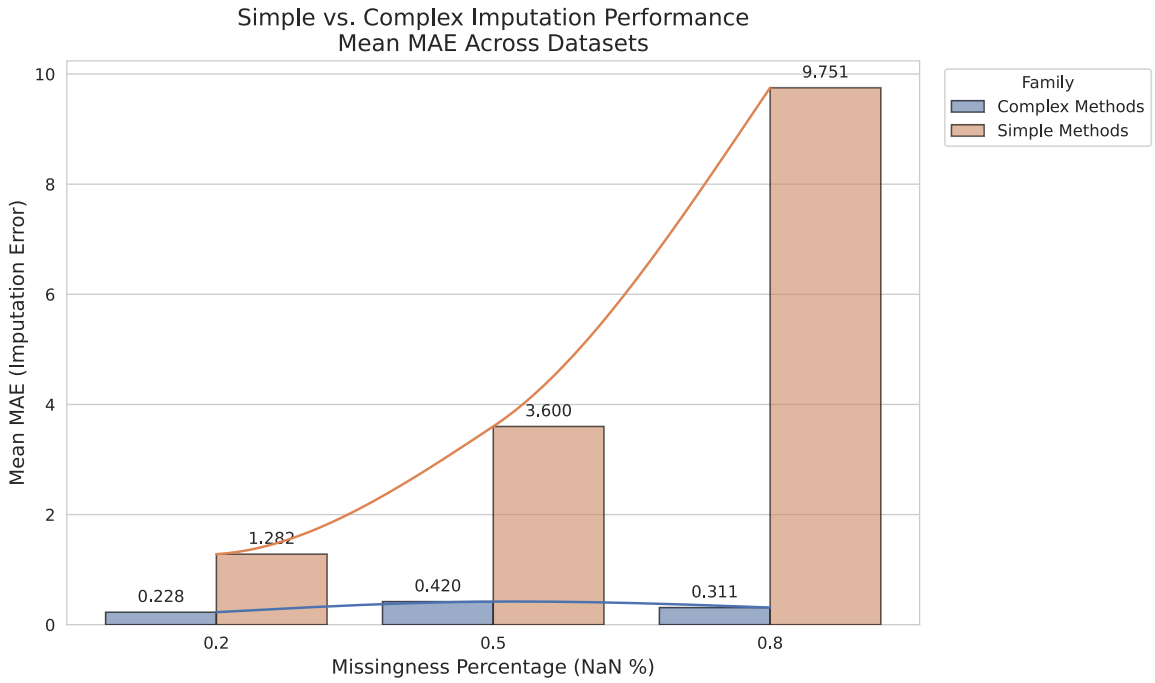


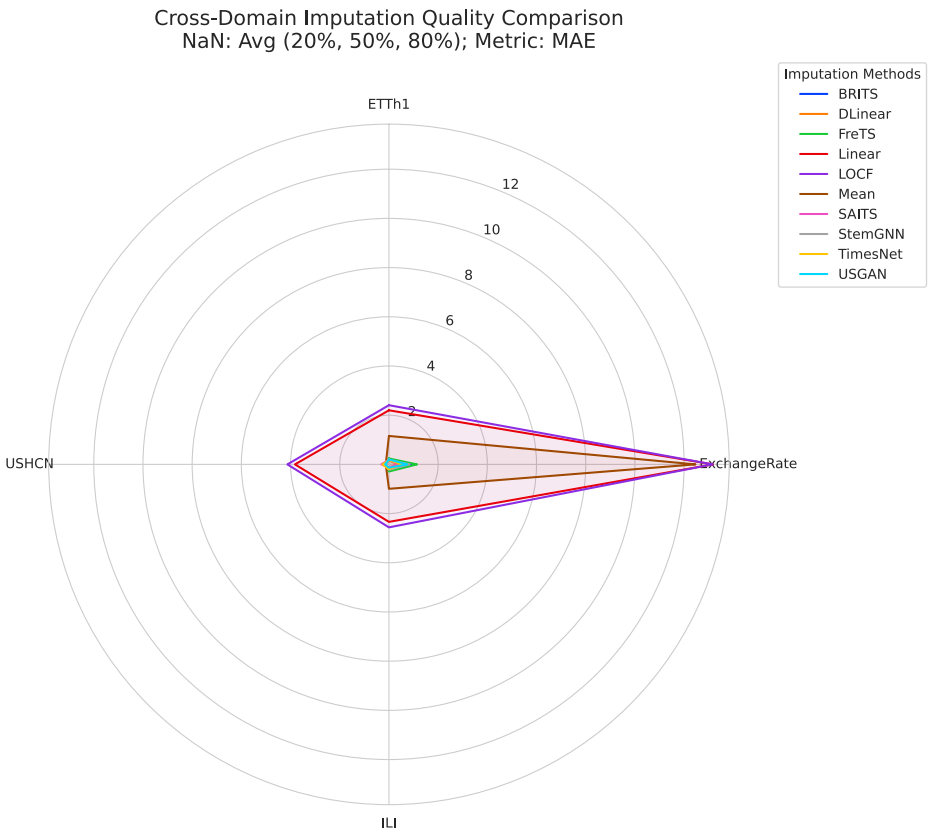
Figure 3.8: Comparison between the two imputation methods families across ETTh1, USHCN, ExchangeRate, and ILI on 20%, 50%, and 80% missing data scenarios.

3.7.2.1 "No Free Lunch" Observations: Architectural Failures and Successes

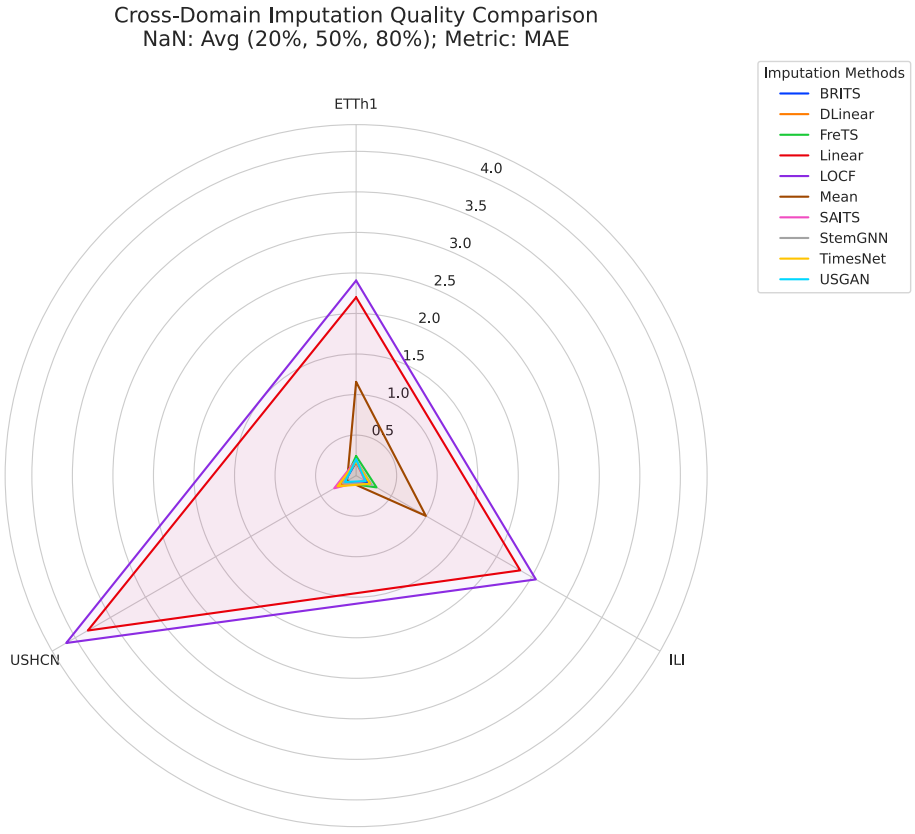
To better study the performance of each imputation method over the chosen datasets, range plots such as Figure 3.9 were analyzed. All imputation methods suffered greatly on the ExchangeRate dataset, and the magnitude difference between the error from the Simple Methods family and the error from the Complex Methods family makes it difficult to accurately estimate the performance of the latter ones. This is confirmed by Figure 3.9i, where it is easier to see the error on the other datasets, but still, it is hard to compare Simple Methods and Complex Methods. Therefore, Figure 3.10 reports the performance of the complex methods only on all datasets in Figure 3.10j, and excluding ExchangeRate in Figure 3.10j. In the latter, it's easier to see that the No Free Lunch Theorem is once again confirmed: even though it might seem like it, there is no single model that achieved the best performance on all datasets, e.g., FreTS was great on USHCN, but had the highest error on ILI and ETTh1; SAITS was the best on ILI, but was the worst on USHCN; USGAN was the best on USHCN, but it was beaten by SAITS on ILI and by StemGNN, DLinear, and BRITS on ETTh1. This finding highlights that no model achieved the best performance on average across different missing data percentage scenarios. Ulterior plots that show the models' performance on specific missing data percentages are reported in Section A.4.3.

3.7.2.2 Statistical Validation Across ISTS Scenarios

Figure 3.11 reports the Critical Difference diagram for all imputation methods across all datasets and missing data percentages. At first glance, the diagram might appear inconclusive. However, the Friedman test confirms a highly significant difference in imputation performance across the evaluated architectures ($p < 0.0001$). The subsequent Nemenyi post-hoc analysis,

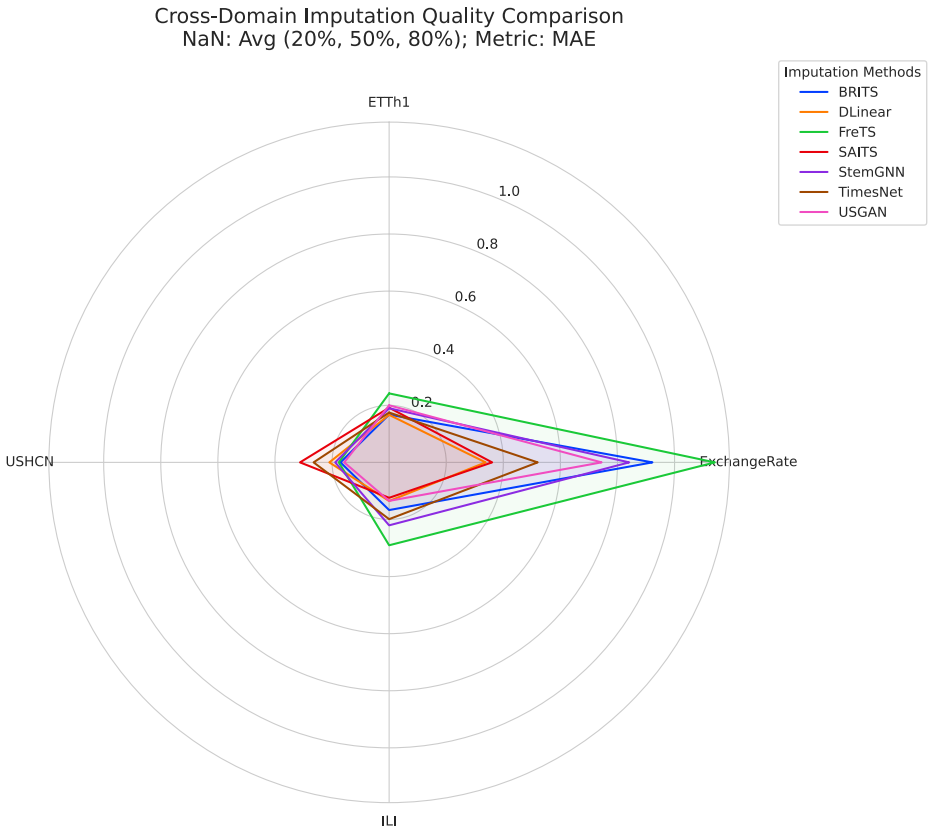


(a)

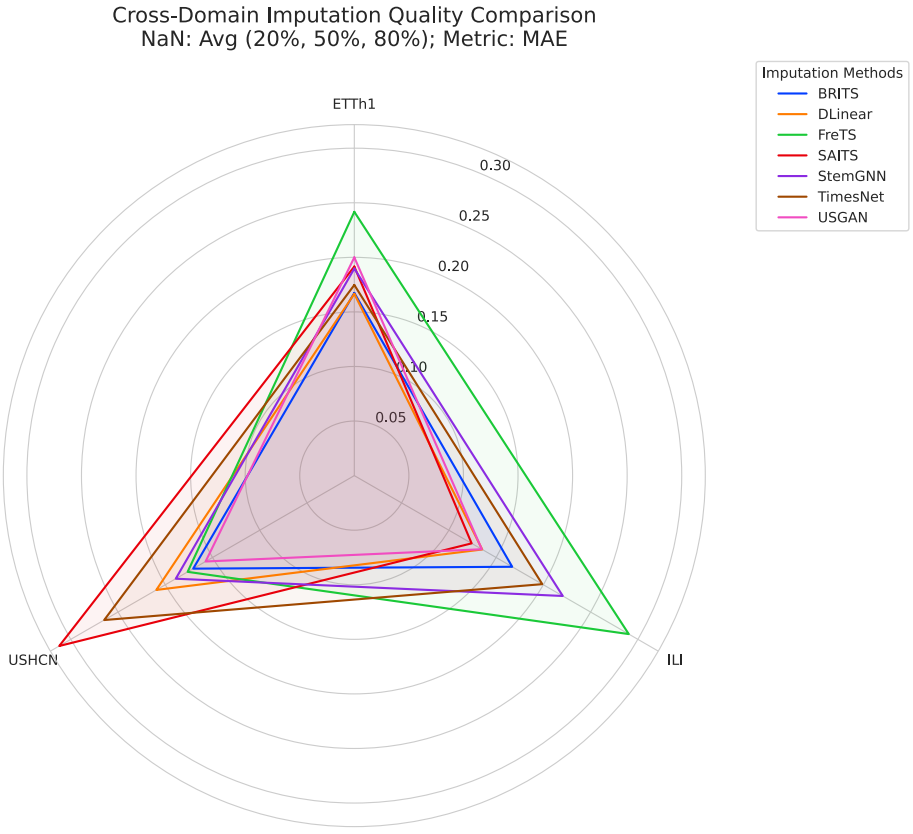


(b)

Figure 3.9: Imputation performance overview of all imputation methods over ETTh1, USHCN, ILI, and ExchangeRate (a), and over all datasets except ExchangeRate (b).



(a)



(b)

Figure 3.10: Imputation performance overview of all complex imputation methods over ETTh1, USHCN, ILI, and ExchangeRate (a), and over all datasets except ExchangeRate (b).

employed to generate the diagram, produces overlapping critical difference cliques. This suggests that while a global performance hierarchy exists — with SAITS and TimesNet consistently occupying the top ranks — the high variance across heterogeneous domains (e.g., the contrast between the periodic ETTh1 and the stochastic ILI) prevents a statistically significant separation between the leading complex models and simple baselines at a confidence level of $\alpha = 0.05$. To better study this phenomenon, Figure 3.12 reports the Critical Difference diagrams on 20%, 50%, and 80% missing data scenarios. In all three plots, the critical difference clique overlaps all models. Therefore, it is assumed that this phenomenon is due to the heterogeneity of the datasets and the use of average metrics over five runs. With a broader view over the models' performance and more data at hand, the Nemenyi post-hoc analysis would probably have been more accurate in distinguishing the statistical relevance of each method. Nonetheless, the position of the methods on the Critical Difference range still highlights relevant information. The Direct Reconstruction analysis identifies a clear performance hierarchy among imputation methods, despite the conservative nature of the post-hoc statistical tests: **complex methods** occupy the top-performing ranks, with **BRITS**, **DLinear**, and **USGAN** consistently achieving the lowest average ranks (ranging from 2.5 to 4.5). SAITS and TimesNet perform the best on the 20% missing data scenario, showing that when missing data cannot be tolerated, they could be worth the extra time and infrastructural cost to fill the dataset with accurate data.

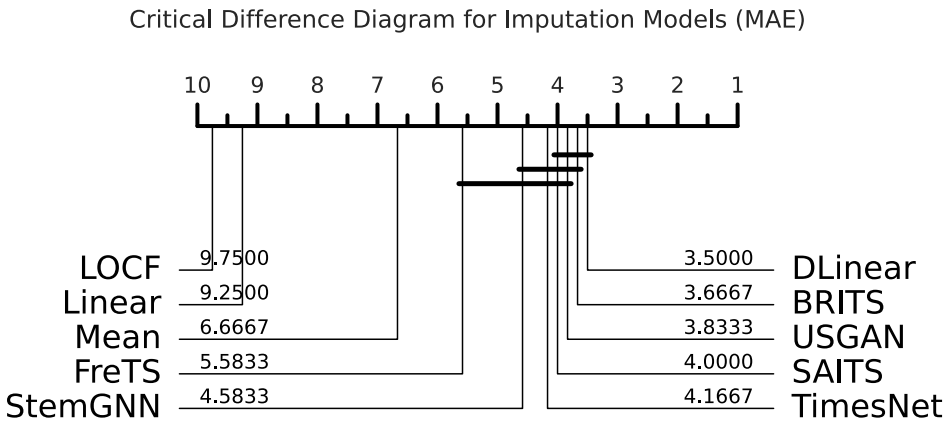
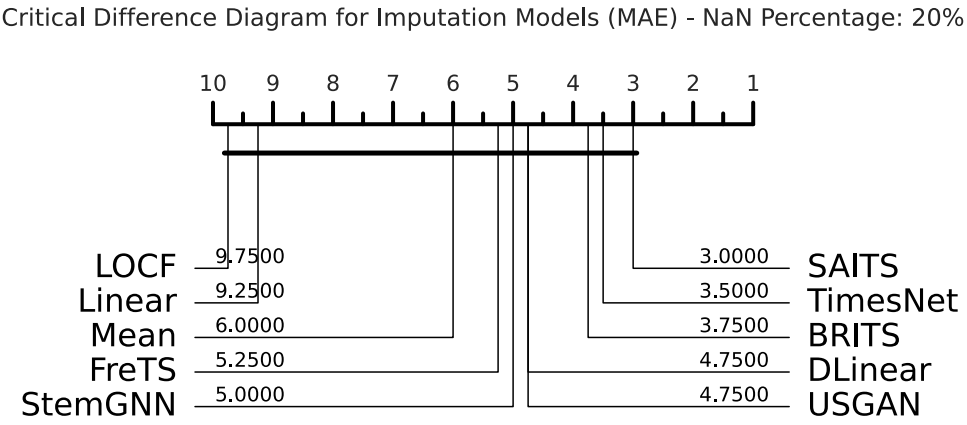


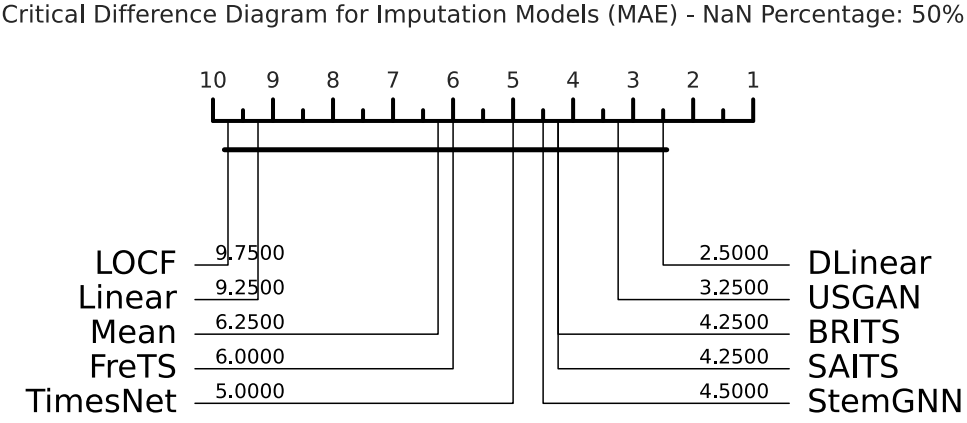
Figure 3.11: Critical Difference diagram for all imputation methods across all datasets and nan percentages.

3.7.2.3 RQ2.1 Summary

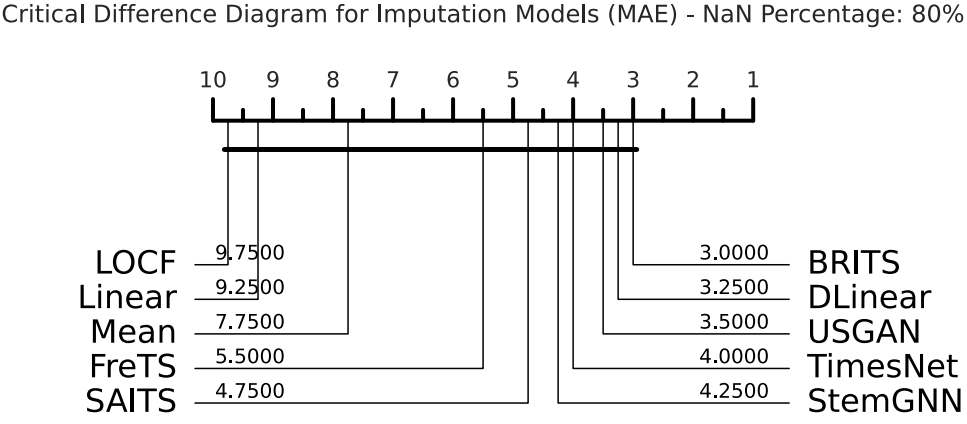
The objective of Research Question 2.1 was to determine which of the ten evaluated approaches provides the most faithful reconstruction of irregularly sampled data across various domains and sparsity levels. The analysis confirms that while complex deep-learning-based imputation consistently outranks simple methods, the choice of the "best" model is highly dependent on the underlying data dynamics.



(a)



(b)



(c)

Figure 3.12: Critical Difference diagrams of all imputation methods for 20%, 50%, and 80% missing data scenarios using MAE as the error metric.

Key Findings.

1. Complex imputation methods achieve significantly lower MAE compared to simple methods (LOCF, Mean, and Linear) across all evaluated scenarios. Simple methods remain a "hard baseline" primarily due to their low computational cost and speed, whereas complex models like SAITS or TimesNet introduce a higher training and inference burden.
2. Complex methods show their greatest utility in the 80% missing data scenario. At this extreme sparsity, where Native ISTS models struggle, deep imputation provides a more effective "learning foundation" for downstream tasks, even if the reconstruction includes some "hallucination".
3. Domain-Driven Performance (No Free Lunch): Architectural success is tied to the dataset. For instance, FreTS excelled on the multi-source USHCN dataset but struggled with ILI and ETTh1. Similarly, SAITS demonstrated superior recovery on the stochastic ILI dataset but was the lowest-performing complex model on USHCN.
4. While the Friedman test confirms highly significant differences between methods ($p < 0.0001$), the Nemenyi post-hoc analysis often produces overlapping cliques. This indicates that the high variance across heterogeneous domains (e.g., periodic vs. stochastic data) prevents a definitive statistical separation between the leading complex models.

The Verdict. Despite domain sensitivity, a clear performance hierarchy exists based on average ranks across all datasets and sparsity scenarios. The Top 3 imputation methods identified for this benchmark are:

1. BRITS
2. DLinear
3. USGAN

These models consistently achieved the lowest average ranks (between 2.5 and 4.5) across the broad spectrum of ISTS scenarios. Additionally, SAITS and TimesNet are highlighted as top-tier choices for scenarios with lower missingness (20%), where their high-fidelity reconstruction justifies the extra infrastructural cost.

3.8 Conclusions, Contributions and Future Work

A core goal of this framework was to eliminate the "SOTA chase" artifacts identified in the literature review (Section 2.1). By utilizing shuffled sliding windows for training, the models had to learn the data's temporal dynamics from different points in time, avoiding the biases and potential overfitting of a single view on the past. The use of multiple random seeds to generate distinct missingness realizations enabled the isolation of models that were "lucky" with a specific gap pattern from those that possess genuine architectural robustness. The strict separation of scaling parameters (derived only from training data) prevented the artificial inflation of accuracy scores often seen in papers that normalize using global dataset statistics. This benchmark provides a comprehensive evaluation of the trade-offs between specialized ISTS architectures

and standard forecasting pipelines. The results indicate that while Native ISTS models (notably PrimeNet) offer superior reliability and a higher "predictive floor" in sparse or high-variance domains, Standard pipelines paired with optimized imputation remain highly competitive in stable or periodic domains. A major contribution of this work is the identification of the "Reliability Gap". Using R^2 as a core metric, it was demonstrated that models can achieve low absolute errors (MAE) while fundamentally failing to learn the temporal signal. Furthermore, our evaluation of imputation methods confirms that while deep-learning approaches, e.g., BRITS, DLinear, and USGAN, provide the most faithful signal reconstruction, their use is most critical at extreme sparsity (80% NaN), where simple baselines like LOCF fail by introducing too much error for a reliable prediction.

This framework was developed from scratch, from the data reading and preprocessing phase to the final forecasting training procedures. The algorithms were optimized, exploiting parallelization where possible, to create a product that is not only effective, but also efficient time-wise and space-wise. The framework allows for the automatic management of a set of GPU devices, making it possible to run multiple experiments simultaneously, exploiting all available hardware. The number of CPU cores and GPU devices can be tweaked according to available resources, so to be compatible with workload schedulers like SLURM typically found in **H**igh **P**erformance **C**omputing (HPC) clusters. The framework was also built with extensibility and maintainability in mind. The software was designed so that new datasets and models can be easily added; functionalities were implemented so that they are easy to understand, read and extend.

As of now, the code is not publicly available, and the research is still ongoing. The collection of such an extensive amount of data — which uses computationally expensive models — requires a proportionate infrastructure, and, consequently, a large amount of time to be completed. Therefore, as soon as the gathered metrics produce conclusive results, the work will be published.

Future work on this framework includes:

1. RQ2.2: Study which of the top 3 imputation algorithms actually improve downstream forecasting.
2. RQ2.3: Study the impact that complex methods have over Imputation + Standard model pipelines (replacing the LOCF strategy) in comparison to Native ISTS forecasting.
3. RQ3: Study the impact that other missingness patterns have on the effectiveness of forecasting models and imputation methods.
4. RQ4: Study the impact of the look-back window size on the different architectures available for forecasting and imputation, varying the forecasting horizon and missingness patterns.
5. RQ5: Study temporal and spatial requirements to quantify exactly when the performance gain of a Complex Imputation or Native ISTS model justifies its high training cost.
6. RQ6: Study how much models actually learned in different scenarios by measuring the forecasting performance over the test set with various levels of data missingness. Good

performance over a test set with a high percentage of missing data means the model has good generalization capability, even with little data to use as a basis for the prediction.

One interesting improvement over the current framework structure would be the development of a "meta-learning" layer that automatically selects the optimal imputation method (e.g., switching from LOCF to SAITS) based on a real-time sparsity scan of the incoming data. This would improve the quality and potential of the software, making it an even better tool for researchers to automate experiments and data gathering.

To conclude this section, Table 3.5 summarizes the findings reported throughout the conducted experiments.

Scenario	**Recommended Strategy**	**Primary Justification**
Low Sparsity (20% NaN)	Standard + Simple Imputation	Standard pipelines like PatchTST paired with LOCF are highly competitive and cost-effective
High Sparsity (80% NaN)	Standard + Complex Imputation	Native ISTS models struggle with data scarcity; complex imputation (e.g., BRITS) provides a necessary "learning foundation"
High Variance Domains	Native ISTS	Architectures like PrimeNet offer superior resilience, maintaining a predictive "floor" (positive R2) where others fail
Periodic/Stable Data	Linear / MLP Models	Simple mappings like DLinear become statistically competitive or superior in stable regimes with extreme sparsity
Trustworthiness Check	Monitor R2 Score	Low MAE can be deceptive; negative R2 identifies "failure regimes" where models fail to capture meaningful dynamics

Table 3.5: Summary of Benchmarking Recommendations.

CHAPTER 4

Discovery in Multidimensional Space: Novel Spatio-Temporal Clustering for Geospatial Phenomena

Persistent Scatterer Interferometric Synthetic Aperture Radar (PS-InSAR) has advanced ground deformation monitoring since its introduction in the early 2000s [FPR00], [FPR01]. This technique builds upon satellite-based advanced interferometric methods by focusing on stable, highly reflective points, known as Persistent Scatterers (PS), to measure surface displacements with sub-millimeter precision over wide areas [Cro+16]. PS-InSAR processing generates time series representing cumulative displacement along the satellite's Line-of-Sight (LOS) direction [Osm+16], with contributions from vertical and east-west components due to limited north-south sensitivity [Han01], [BH21]. LOS measurements are acquired from ascending and descending satellite passes, each providing unique perspectives on ground deformations. Combining the ascending and descending LOS measurements offers a powerful approach to vector decompose the observed deformations [CA21]. This decomposition results in two distinct time series for each PS: one representing vertical displacements (along the up-down axis; UD) and another representing horizontal displacements (along the east-west axis; EW). This separation of components facilitates a more comprehensive and subtle analysis of ground deformation patterns that often manifest as a complex interplay of vertical and horizontal movements. Consequently, for phenomena such as landslides and subsidence [Ant+19], [Gha+24a], the resolved vertical and horizontal components enable a deep understanding of the spatiotemporal dynamics of displacements.

While PS-InSAR offers sub-millimeter precision over wide areas for ground deformation monitoring [Guz+12], [Maz+22], it also presents significant challenges in data analysis and interpretation, since the number of time series often ranges from millions to billions of points across a given area of interest [Sou+10], [MBM21], doubling if vector decomposition is considered. Consequently, efficiently analyzing and extracting meaningful ground movement information from these extensive amounts of data has become increasingly challenging and labor-intensive, necessitating the development of advanced data processing and interpretation techniques [TS24]. Clustering, in particular, has emerged as a powerful technique for identifying patterns and grouping similar time series [ASY15]. Recent advancements in PS clustering have focused on developing unsupervised tools and methods to analyze large volumes of data across various scales and applications [SS21], [ZZZ22], [Jin+23], [Fes+23], [Pen+24], [Zoc+25]. Several approaches in this field directly analyze raw LOS displacement time series. [SS21] employed a DBSCAN (Density-Based Spatial Clustering of Applications with Noise) with a hybrid distance metric to group PS exhibiting similar temporal deformation patterns at the building scale. [ZZZ22], [Jin+23] applied a K-Shape clustering algorithm for local-scale subsidence analysis, capturing shape-based similarities

in temporal profiles. Addressing large-scale interferometric data, [Pen+24] employed a distance-based K-Means analysis to interpret extensive spatial patterns. However, these methods rely on the direct use of raw time series, which presents several challenges. Noisy or low-quality time series can compromise data integrity, leading to information loss and less reliable clustering results. Furthermore, processing long or large datasets is computationally intensive, making these approaches less efficient for large-scale applications. Complementing the LOS-based approaches, [Fes+23] introduced a K-Means clustering that separately processes the up-down and east-west components to identify displacement patterns at a regional scale, using Principal Component Analysis (PCA) as a dimensionality reduction technique for feature extraction. However, PCA extracts features that reside in a latent space, making them inherently uninterpretable for users. This lack of explainability limits the applicability of this approach when trying to understand which specific characteristics are important for predicting the displacement patterns.

While current investigations demonstrate promising results, their validation relies primarily on traditional unsupervised metrics like Silhouette score [MS11], [TS24], Dunn index [Chi+24], or internal validation measures [Fes+23]. These conventional techniques assume well-defined spherical clusters of uniform density [SN20], whereas ground deformation exhibits irregular distributions and complex patterns with varying densities and shapes, often lacking distinct cluster boundaries. These spatial characteristics fundamentally challenge traditional clustering assessments, as neighboring regions may simultaneously represent distinct deformation mechanisms or transitional zones with gradual displacement intensity gradients, rendering standard metrics insufficient for comprehensive PS clustering analysis. Beyond these evaluation limitations, several other challenges persist in current approaches: i) conventional raw-based approaches may fail to capture complex temporal patterns and relationships in time series data; ii) PCA effectively reduces data complexity but does not inherently perform a comprehensive data mining process capable of revealing meaningful, interpretable information from the time series; iii) the studies do not integrate geospatial features, such as point coordinates and other topographic informative attributes, into the time series clustering process; iv) the employment of univariate methods to group similar displacements generally overlooks potential correlations between horizontal and vertical displacement components. These limitations highlight the need for more interpretable multivariate clustering techniques and spatially-aware evaluation techniques that can leverage the full potential of PS data, providing deeper insights into complex deformation processes.

This study, carried on by Claudia Masciulli, Giacomo Guiduzzi, Donato Tiano, Marta Zocchi, Francesco Guerra, Paolo Mazzanti, and Gabriele Scarascia Mugnozza [Mas+25] proposes a comprehensive data-driven approach based on feature extraction to address the limitations of current PS-InSAR clustering techniques. Using the K-Means methodology established by [Fes+23] as a benchmark, we evaluate and compare the effectiveness of shape- and feature-based clustering methods. This framework explores the potential of interpretable approaches that extract meaningful characteristics from time series data. Furthermore, we extend our analysis to simultaneously handle vertical and horizontal displacement components through multivariate clustering techniques to enable a more complete characterization of ground deformation patterns. Our methodology incorporates an adaptive optimization approach for cluster number determination and introduces a tailored evaluation score designed to handle the spatial characteristics of PS data during clustering quality evaluation. The framework was tested and validated in the Offida municipality of the Marche region (Italy), where complex slope deformation patterns offered an ideal testing site for comparing clustering performances.

The contribution of this work lies in addressing key limitations of existing clustering methods for PS-InSAR data analysis by introducing a framework tailored to handle the multi-component displacements and geospatial attributes of such data. Specifically, this research: (i) develops an online optimization approach for determining the optimal number of clusters, overcoming the limitations of current methods that require specifying a maximum number of clusters (k), which is impractical in the PS-InSAR context due to the difficulty of defining an exact maximum number of deformation patterns; (ii) implements a feature engineering framework that extracts interpretable temporal patterns and spatial relationships from PS data, enhancing explainability and enabling users to understand the importance of the selected features; (iii) introduces an adaptive clustering methodology that integrates multiple displacement components while preserving their physical meaning, addressing the gap in methods that fail to account for the complexity of multi-component displacements; (iv) incorporates geospatial attributes to improve the recognition of deformation patterns, bridging the gap between spatial and temporal analysis; and (v) develops a customized evaluation score that reflects the unique characteristics of PS data, ensuring a more accurate and meaningful assessment of clustering performance.

This contribution aligns with current trends in machine learning by combining feature extraction and multivariate analysis to characterize complex spatiotemporal displacement patterns while emphasizing interpretability and advancing automated surface displacement analysis.

4.1 Materials

4.1.1 Study Area

The study was conducted in the Offida municipality, located in the Marche region of Italy, covering an area of approximately 50 km^2. The territory is characterized by moderate relief energy, with elevations ranging from sea level in the main valleys to approximately 500 m near Offida town. The geological setting is characterized by Plio-Pleistocene formations composed of alternating soft and hard terrigenous lithotypes, predominantly marls and pelitic-arenaceous successions [PDG13], [Mar+15], [Buc+21]. The combination of lithological, morphological, seismic, and climatic conditions makes the area particularly susceptible to mass movements. According to the Hydrogeological Asset Plans (PAI) inventory, the predominant slope processes include slides, earth flows, and mud flows, which frequently affect urban settlements [Ama+20], [Mam+23], [Gha+24b]. Due to the frequent occurrence of seismic events and prolonged rainfall that trigger new movements or reactivate existing ones [Per+12], [Don+23], the actual boundaries of landslides may differ significantly from those documented in the inventory, reflecting the dynamic morphological evolution of these phenomena.

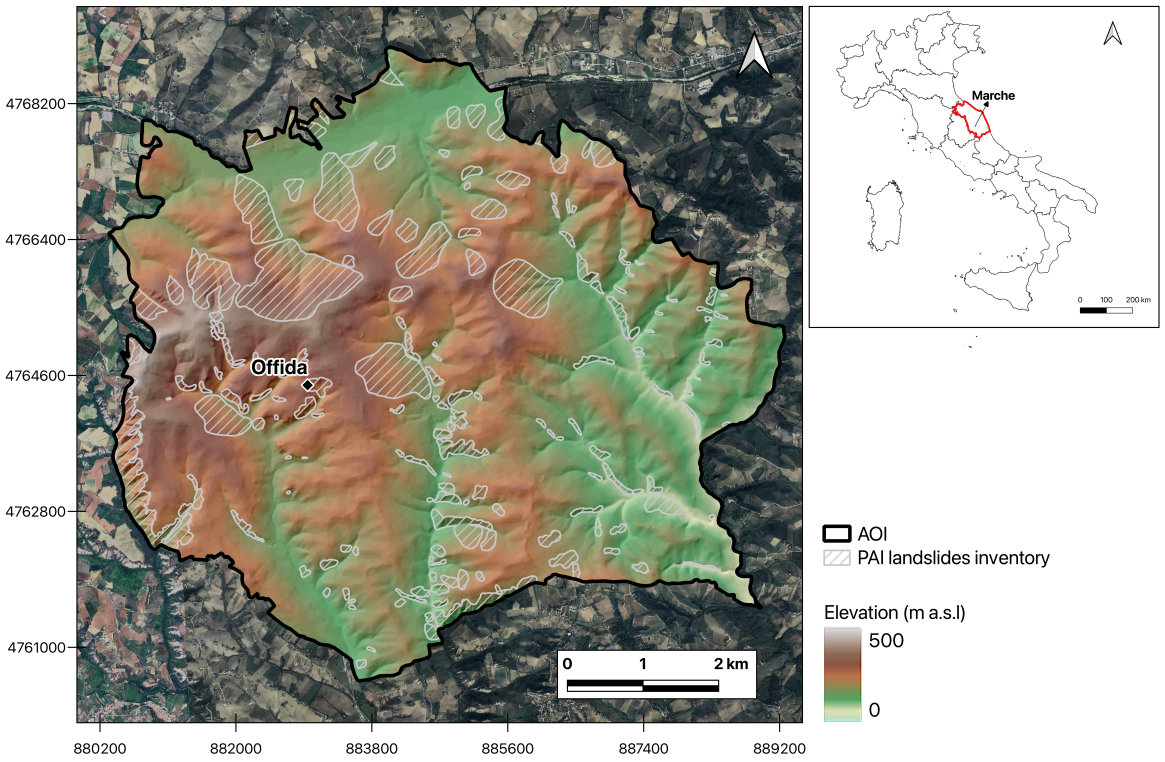


Figure 4.1: Overview of the study area, including the municipality of Offida. White polygons delineate the boundaries of known landslides from the PAI inventory. EPSG:32632

4.1.2 Input Datasets

A summary of key datasets employed in this study is reported in Table 4.1.

Dataset	Spatial Resolution	Description	Source
PS	20x5 meters	ASC/DESC time series	https://egms.land.copernicus.eu/ EGMBS Basic - (Level 2)
DEM	10 meters	TINITALY	https://tinitaly.pi.ingv.it/
PAI	N/A	Landslide inventory	https://aubac.it/piani-di-bacino/piani-di-assetto-idrogeologico

Table 4.1: References of input datasets elaborated in this study.

The primary inputs for our analysis are the PS data derived from the European Ground Motion Service (EGMS), a comprehensive initiative within the Copernicus program that provides standardized and freely accessible PS across Europe [Cro+20], [Cos+21], [Cro+21]. Specifically, we used EGMS basic (Level 2) products, consisting of ascending and descending time series covering the period 2018-2022 (https://doi.org/10.2909/1b2eb0d6-8c3a-4de2-b99d-10a30079f3cc, Figure Figure 4.2). These PS data result from the interferometric analysis of C-Band Sentinel-1 images processed at full resolution, using a virtual local reference point.

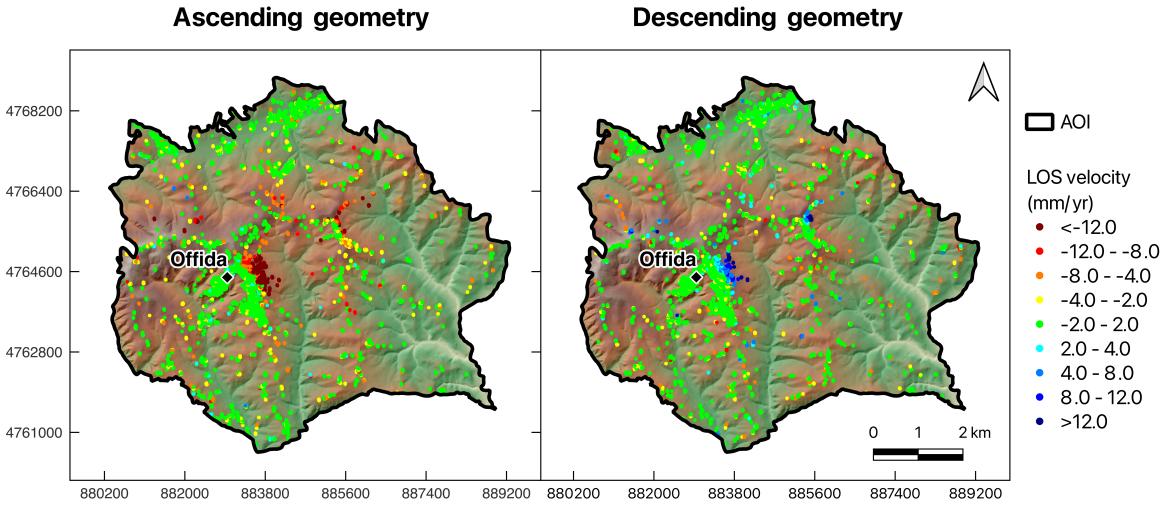


Figure 4.2: Spatial distribution of PS points in the study area. (left) Ascending orbit PS data; (right) Descending orbit PS data. Colors represent LOS velocity in mm/year, with negative values indicating movement away from the satellite and positive values displaying movement towards the satellite. EPSG:32632

The main characteristics of the datasets are provided in Table 4.2.

Sentinel-1 PS characteristics		
Orbital geometry	Ascending	Descending
Spatial resolution [m]	5x20	5x20
Track	044	022
Number of images	269	261
Incident angle [°]	36.32	39.51
Heading angle [°]	−9.50	190.81
Number of PS	13449	12278

Table 4.2: Summary of the main characteristics of the Sentinel-1 datasets.

To complement the PS data analysis, we incorporated additional datasets that provide context for the local topography and known landslide occurrences. We employed the TINITALY Digital Elevation Model (DEM) [Tar+07], [TN17], [Tar+23], which provides high-resolution topographic data for the entire Italian territory at a spatial resolution of 10 meters. From this DEM, we derived additional terrain parameters including slope, aspect, curvatures, Terrain Ruggedness Index (TRI), and Topographic Wetness Index (TWI). These parameters characterize the complex terrain features that may influence landslide occurrence and behavior [Fes+22]. Additionally, we integrated data from the PAI landslide inventory, developed by regional authorities, which documents 174 landslides covering an area of 8.3 km^2 within our study region. This inventory provides valuable reference information for understanding the spatial distribution of known slope displacements.

4.1.3 PS Post-processing

To ensure the quality and reliability of our analysis, the EGMS dataset underwent a two-step filtering pipeline: i) noise reduction using a temporal coherence threshold of 0.6, preserving over 90% of relevant information, and ii) removal of isolated points [Mil+19], [Sel+20], [FAT21]. Isolated points were removed by implementing a spatial criterion based on typical gravity-driven phenomena (average size 0.05 km^2), requiring at least three neighboring points within a 50-meter radius of each PS [Bar+17], [Sol+19].\ To estimate the two-dimensional components of ground motion along the vertical and east-west horizontal directions, we applied vector decomposition (Figure Figure 4.3) to the filtered PS from ascending and descending orbits.

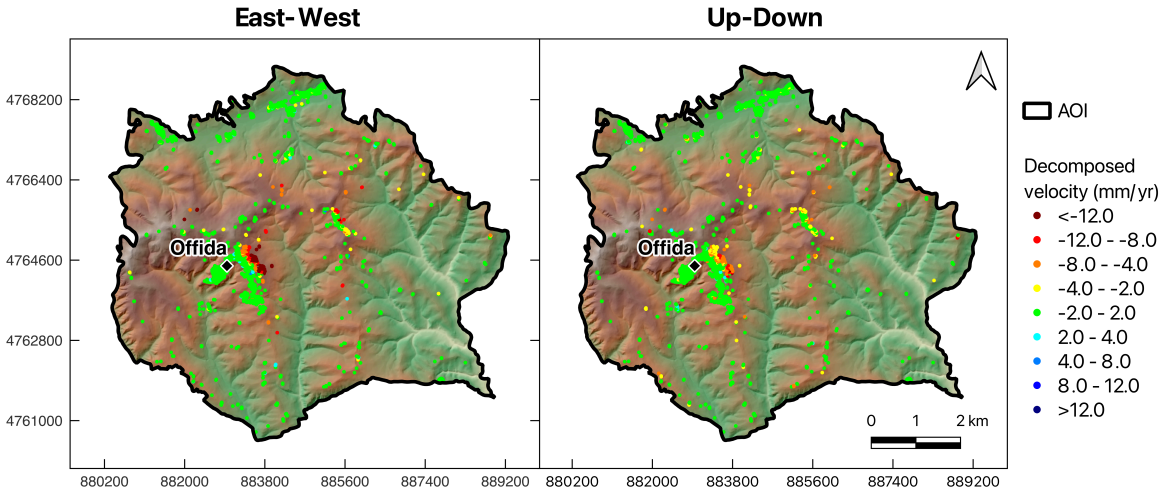


Figure 4.3: Decomposed PS data in the study area. East-west horizontal displacement rates (left) and vertical displacement rates (right). Colors represent the decomposed velocity in mm/ year, with negative values indicating westward or downward movement, while positive values indicating eastward or upward movement, respectively. EPSG:32632

The study area was discretized into a 10x10 meter hexagonal grid, enabling the combination of PS points from both orbital geometries [Bis+20]. At each corner of the hexagonal cells, we established a point if both ascending and descending orbits provided valid measurements, ensuring a more comprehensive and accurate representation of ground motion across the study area. Following the methodology outlined by [Fes+23], the vertical (V_y) and horizontal (V_x) time series for each point are obtained by solving the following system of equations:

$$\begin{pmatrix} v_a \\ v_d \end{pmatrix} = \begin{pmatrix} -\sin\theta_a \cos\alpha_a & \cos\theta_a \\ -\sin\theta_d \cos\alpha_d & \cos\theta_d \end{pmatrix} \begin{pmatrix} V_x \\ V_y \end{pmatrix} \tag{4.1}$$

where v_a and v_d represent the LOS velocities for ascending and descending geometries. The incidence ($\theta_{a,d}$) and heading ($\alpha_{a,d}$) angles for a given satellite and geometry are reported in Table 4.2. Assuming a negligible displacement along the north-south component, the total displacement vectors can then be calculated as the point-by-point vectorial sum of the 2D displacement components [Bis+20], [CA21]:

$$V_{tot} = \sqrt{V_y^2 + V_x^2} \tag{4.2}$$

To illustrate the decomposition process and the calculation of total displacement, Figure Figure 4.4 presents an example of decomposed time series data for a representative PS point in our study area.

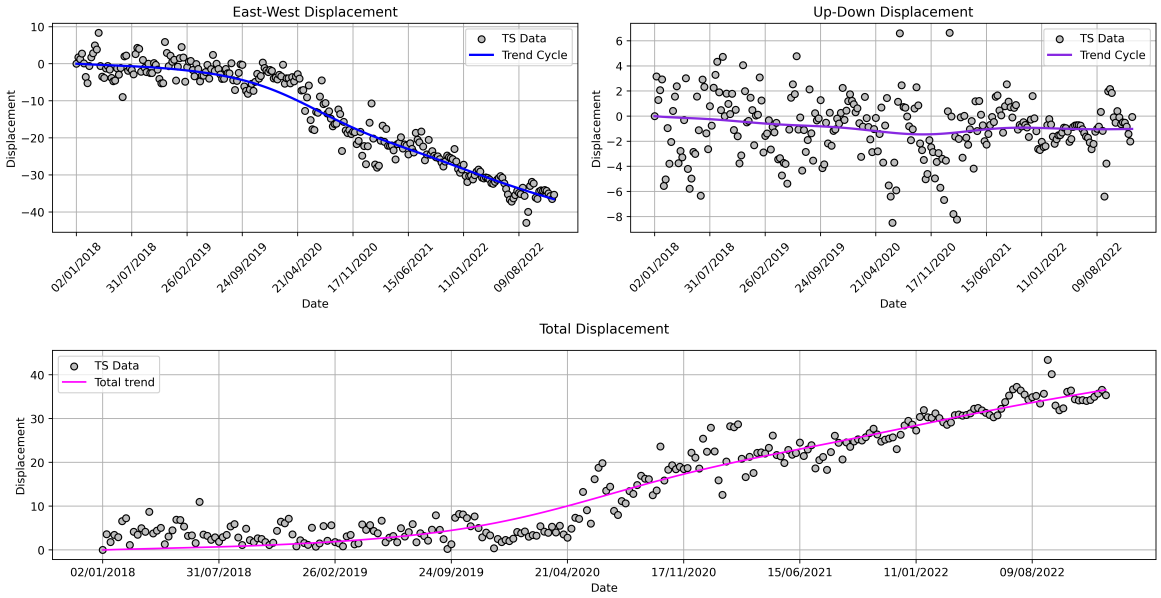


Figure 4.4: Decomposed time series and total displacement for a representative PS point (id: 968). Top left: east-west displacement component. Top right: up-down displacement component. Bottom: Total displacement given by combined displacement trends. The long-term trends are extracted using an additive seasonal trend decomposition based on a locally weighted regression smoother.

4.2 Methods

This study introduces a framework comparing clustering techniques for PS time series analysis. The methodology integrates feature engineering, spatial attributes, and both univariate and multivariate approaches, employing raw time series-based algorithms (K-Means and K-Shape) and feature-based techniques (FeatTS and Time2Feat) to detect deformation patterns in landslide-prone areas.

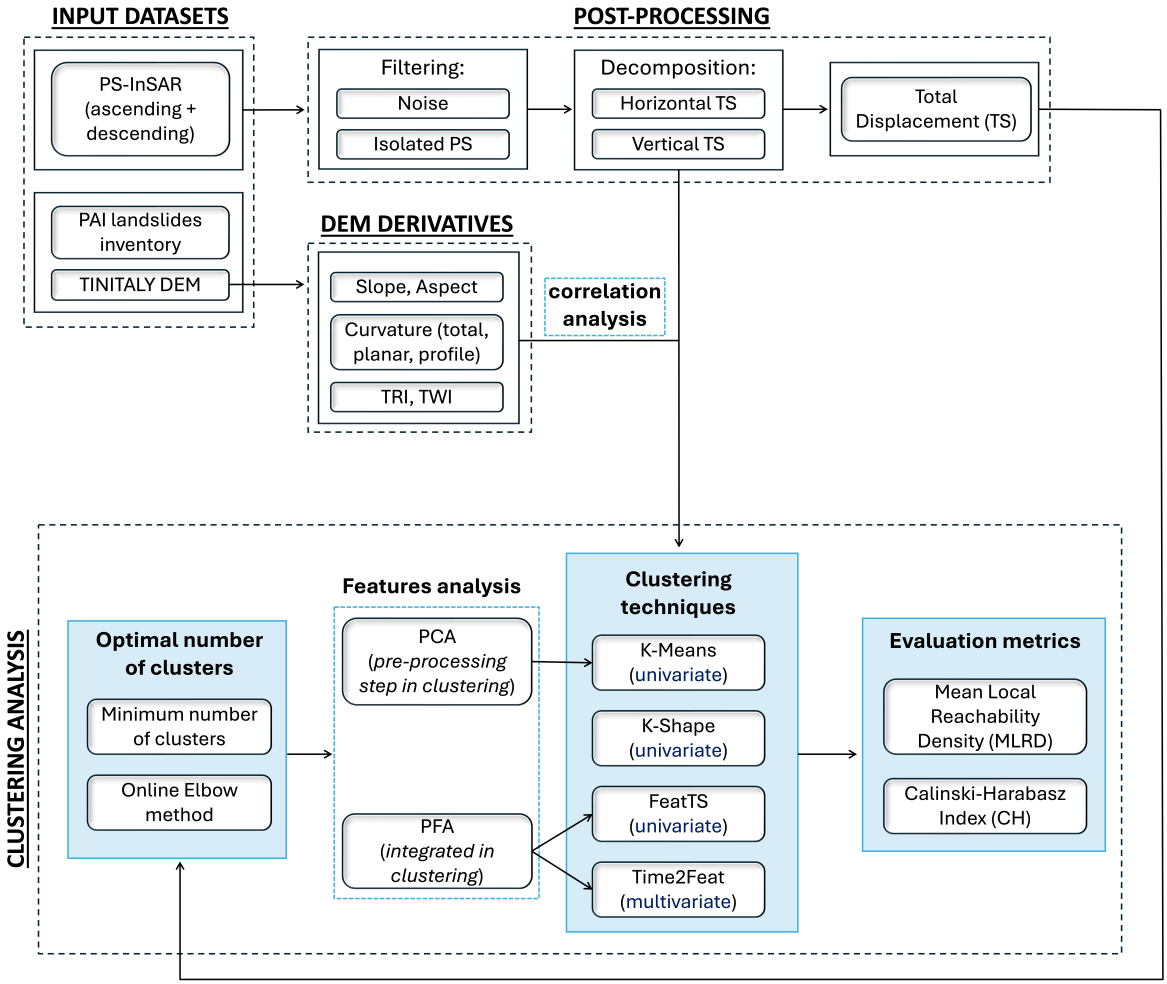


Figure 4.5: Schematic overview of the analytical framework applied in this study.

Figure 4.5 illustrates the workflow. The methodology processes decomposed displacement components, total displacement time series, and geospatial features from Section 4.1.2 and Section 4.1.3. We first determine the optimal number of clusters using the online approach detailed in Section 4.2.1. Using this cluster size, we apply the unsupervised clustering techniques outlined in Section 4.2.2. For methods leveraging external information and time series characteristics, we implement a two-fold feature analysis that combines geospatial correlation analysis with temporal pattern extraction through interpretable data mining techniques (Section 4.2.3). The performance of each clustering approach is then evaluated using specialized metrics to assess their capability in capturing complex dynamics (Section 4.2.4). The validation framework compares clustering results against baseline displacement patterns derived directly from PS displacement measurements. These baseline patterns, detailed in Section Section 4.2.5, provide a physically interpretable reference that enables objective assessment of clustering performance, particularly for identifying deformation zones beyond mapped landslide boundaries. This structured approach enables systematic evaluation of clustering strategies, providing insights into the comparative advantages of feature-based methods for PS analysis.

4.2.1 Cluster Number Optimization

The selection of an appropriate number of clusters is critical in unsupervised learning. Traditional methods, like the elbow method, require a predefined range of cluster numbers

and the computation of inertia values, which can be resource-intensive for large time series datasets. These strategies often underestimate the optimal cluster count, which is particularly problematic in analyzing ground deformation patterns where multiple displacement phenomena can occur simultaneously. To address these limitations, we developed an online optimization algorithm that dynamically determines cluster quantities through adaptive thresholds derived from inertia variations. This automated approach eliminates arbitrary upper bounds on cluster numbers, allowing for iterative identification of inherent data structures within interferometric measurements, ensuring computational efficiency and comprehensive exploration of the solution space.

The algorithm starts with a minimum cluster count (n_{min}) based on the number of landslide polygons containing at least three PS points, preventing underestimation of deformation patterns. The iterative process evaluates clustering quality by monitoring inertia changes as cluster numbers increase, determining the absolute minimum in the inertia distribution. We utilize a memory-efficient mini-batch variant of the K-Means algorithm to enhance computational efficiency, processing small, randomly selected batches of data in each iteration. For each clustering solution with k clusters, the inertia (within-cluster sum of squares) is calculated as:

$$I_k = \sum_{x \in c_i} \|x - \mu_i\|^2 \tag{4.3}$$

where I_k is the total within-cluster variance, c_i is the set of points in cluster i, and μ_i is the centroid of cluster i. The inertia I_k tends to decrease as k increases. The algorithm evaluates the change in inertia between successive clustering solutions to determine the optimal number of clusters. The average change in inertia, ΔI_{avg}, is calculated as:

$$\Delta I_{\text{avg}_n} = \frac{1}{n-1} \sum_{i=1}^{n} (I_{i-1} - I_i) \tag{4.4}$$

where n is the current number of evaluated clusters, and I_i is the inertia of the i-th solution. In each iteration, the decrease in inertia, $\Delta I_k = I_{\{k-1\}} - I_{\{k\}}$, is compared against a dynamic threshold defined by a user-specified parameter, θ:

$$\Delta I_k > \left(\Delta I_{\text{avg}_{n-1}} \cdot \theta\right) + \Delta I_{\text{avg}_{n-1}} \tag{4.5}$$

The parameter θ adjusts the algorithm's sensitivity to changes in inertia. A higher θ prioritizes larger decreases, typically found at lower k, while a lower θ considers smaller decreases, potentially leading to an optimal k at higher values.

Parameter	Value	Description	Rationale
θ	0.1	Sensitivity threshold	Sets the minimum acceptable inertia decrease, relative to the average.
n_{min}	6	Minimum number of clusters	Based on the number of landslide polygons covered by at least three PS points.
patience	100	Maximum consecutive iterations	Ensures thorough exploration of clustering solutions while maintaining efficiency. Reset each time a minimum is reached.

Table 4.3: Summary of parameters used in the online optimization approach.

The search for the optimal number of clusters continues until no improvement in inertia decrease (ΔI_k) is observed for a user-defined number of consecutive iterations, referred to as *patience*. The optimal cluster count is identified where the difference between observed and expected inertia decreases reaches its maximum, indicating substantial improvement in cluster cohesion. This online algorithm enables incremental data processing, efficiently handling large volumes of PS time series without requiring simultaneous processing of the entire dataset. The analysis employs total displacement time series that captures the vector sum of horizontal and vertical movements to determine the optimal cluster count. This approach ensures a comprehensive representation of the ground deformation patterns by detecting the full vector of displacement rather than isolated directional components. By optimizing clusters based on total displacement, we account for the complex interplay between different movement directions that characterize slope instability processes, thereby enabling more accurate identification of coherent deformation mechanisms. The threshold θ is selected based on empirical testing, with values above 0.1 consistently yielding the same number of clusters, effectively capturing significant changes in inertia reduction. The *patience* parameter ensures a thorough exploration of potential clustering solutions, allowing the algorithm to identify robust patterns before convergence.

4.2.2 Unsupervised Clustering Techniques

We evaluated multiple unsupervised clustering techniques differing in their temporal pattern recognition and ability to incorporate external variables [ASY15]. The analysis compares standard K-Means and its PCA-enhanced variant for dimensionality reduction against specialized time series algorithms: K-Shape, which focuses on shape-based pattern recognition, and feature extraction methods FeatTS and Time2Feat, which derive temporal characteristics from displacement data. To ensure consistency, our online optimization approach was applied to maintain a consistent optimal cluster count across all methods and displacement components.

4.2.2.1 Raw-based Clustering

As baseline approaches, we implemented two univariate clustering techniques operating directly on raw time series: K-Means and K-Shape. K-Means groups displacement measurements based on Euclidean distance, minimizing within-cluster inertia (Equation 4.6):

$$\underset{C}{\operatorname{argmin}} \sum_{i=1}^{k} \sum_{x \in C_i} \|x - \mu_i\|^2 = \underset{C}{\operatorname{argmin}} \sum_{i=1}^{k} |C_i| \operatorname{Var} C_i \tag{4.6}$$

where $C = \{C_1, ..., C_k\}$ represents the set of clusters, and μ_i is the centroid of cluster i. Following [Fes+23], K-Means was implemented on raw time series and after PCA-based dimensionality reduction.

K-Shape clustering, designed for time series data, focuses on shape-based similarity while accounting for scale and shift invariance [PG15]. It uses an iterative centroid-based refinement process similar to K-Means but employs the Shape-Based Distance (SBD) metric to quantify similarity (Equation 4.7):

$$\mathrm{SBD}(\boldsymbol{x}, \boldsymbol{y}) = 1 - \max\left(\frac{\mathrm{CC}(\boldsymbol{x}, \boldsymbol{y})}{\sqrt{\mathrm{CC}(\boldsymbol{x}, \boldsymbol{x})[0] \cdot \mathrm{CC}(\boldsymbol{y}, \boldsymbol{y})[0]}} \right) \tag{4.7}$$

where $\mathrm{CC}(\boldsymbol{x}, \boldsymbol{y})$ is the normalized cross-correlation sequence, and $\mathrm{CC}(\boldsymbol{x}, \boldsymbol{x})[0]$, $\mathrm{CC}(\boldsymbol{y}, \boldsymbol{y})[0]$ are the auto-correlations at lag 0.

4.2.2.2 Feature-based clustering

Advancing beyond raw-based clustering techniques, we incorporated feature-based approaches that translate temporal data into interpretable characteristics reflecting the underlying physics of processes. These characteristics include linear trends, seasonal patterns, acceleration phases, and displacement rate variations [Hu+24]. This methodology integrates additional variables, such as geospatial attributes, enabling a comprehensive analysis of both temporal evolution and spatial context of ground deformation patterns. Our implementation includes two complementary approaches: FeatTS (Feature-based Time Series clustering, [TBN21a], a univariate method leveraging feature extraction and graph theory, and its multivariate extension, Time2Feat [Bon+22], [Bon+23].

The FeatTS algorithm operates in two steps. First, it extracts intra-signal characteristics ($\boldsymbol{f} = \boldsymbol{f}_{\text{intra}}$) from each time series, capturing intrinsic properties of individual signals [CKF16], [Chr+18] and creating a multidimensional representation of underlying patterns. Second, clustering is performed using a graph encoding approach [BP09], [LLY10], where time series are represented as nodes in a network, with weighted hyperedges connecting similar series [Leo20]. Graph partitioning is then applied to identify clusters of highly interconnected nodes, effectively grouping time series with similar characteristics [Sch07].

Time2Feat extends this methodology to multivariate time series (MTS) clustering through an end-to-end machine learning system. Building on FeatTS, it integrates inter-signal features ($\boldsymbol{f}_{\text{inter}}$) with intra-signal analysis, quantifying pairwise relationships between signals [Liu+24b]. This dual characterization ($\boldsymbol{f} = [\boldsymbol{f}_{\text{intra}}, \boldsymbol{f}_{\text{inter}}]$) provides a comprehensive representation of both individual patterns and component interactions. For cluster identification, hierarchical clustering is employed, constructing a dendrogram based on the predefined number of clusters [RGP08]. This hierarchical structure enables multi-scale analysis, revealing patterns at various levels of granularity in the MTS dataset [Jol18].

4.2.3 Analysis of Key Features

Feature analysis involved selecting uncorrelated external features and extracting temporal characteristics, analyzing geospatial attributes through Pearson and Spearman correlation coefficients [HK11]. Feature selection among correlated pairs (threshold: 0.8, [SBS18]) was guided by domain knowledge to ensure relevance to landslide processes and remove redundancy.

Temporal feature extraction follows a dual approach for univariate and multivariate analyses. We used the TSFresh library for univariate analysis to compute intra-signal features, such as central tendency, dispersion, entropy, and spectral properties [Chr+18]. For multivariate analysis, inter-signal features were added, quantifying relationships between vertical and horizontal displacement components [Bon+22], such as correlation measures and mutual information [Jas+22]. The resulting high-dimensional feature set was reduced using Principal Feature Analysis (PFA), which selects the most informative features while maintaining interpretability [Lu+07].

4.2.3.1 Principal Feature Analysis

PFA combines feature selection with a variance-based mechanism to identify influential features while preserving their physical interpretability [SGM10]. Unlike PCA, which transforms features into uncorrelated components, PFA selects a subset of original features. The key steps of PFA are:

1. Standardization of input features to zero-mean vectors;
2. Eigendecomposition of the covariance matrix;
3. Projection of features into a lower-dimensional subspace;
4. K-Means clustering of feature centroids;
5. Selection of representative features maintaining a target explained variance threshold.

The covariance matrix is defined as:

$$\text{corr}(X) = \text{V}\Lambda\text{V}^T \tag{4.8}$$

where $X = \{x_1, ..., x_n\}$ represents standardized features, Λ contains eigenvalues ($\lambda_1 \geq ... \geq \lambda_n$), and V contains orthonormal eigenvectors. Rows of V project features into a subspace $\tilde{X} \subseteq X$, where highly correlated features have similar weight vectors [Lu+07]. K-Means clustering groups features based on weight vector similarities, and PFA selects representative features from each cluster based on cumulative explained variance [TBN21b].

Thresholds from 10% to 90% were tested to balance dimensionality reduction and information preservation. Higher thresholds retain detailed physical interpretations, while lower thresholds achieve greater dimensionality reduction while preserving key deformation characteristics.

4.2.4 Performance Evaluation Metrics

Ground deformation pattern evaluation poses unique challenges due to irregular spatial distributions, variable point densities, and intensity gradients in PS measurements [Ryg+23], [GMG24]. Traditional clustering metrics, such as the Silhouette, Dunn, and Davies-Bouldin indices, are

unsuitable for these patterns as they assume well-separated, compact clusters with uniform densities. Their limitations include sensitivity to noise, inability to handle gradual transitions, and misrepresentation of spatial coherence in deformation zones. The limitations of these metrics can be summarized as follows:

- **Silhouette Index**: Assumes well-separated clusters with uniform densities, making it ineffective for handling density variations and gradual transitions in ground deformation zones.
- **Dunn Index**: Relies on extreme distance calculations, making it overly sensitive to noise and spatial variations, and unsuitable for continuous, gradually varying deformation patterns.
- **Davies-Bouldin Index**: Favors compact, balanced clusters, misrepresenting the quality and spatial coherence of deformation patterns with varying sizes and intensity gradients.

While the Calinski-Harabasz (CH) index shows some adaptability by evaluating variance-based cluster quality, it still favors similarly-sized clusters, limiting its effectiveness for deformation zones with varying extents. To address these limitations, we propose a custom evaluation score that integrates point density assessment with feature-based similarity measures. This score focuses on the cohesion of deformation zones while preserving the independence of distinct displacement phenomena. It builds upon the Local Reachability Density (LRD), quantifying a point's local density relative to its k-nearest neighbors. The LRD is calculated as:

$$\mathrm{LRD}_k(\boldsymbol{x}) = \frac{1}{\frac{1}{|N_k(x)|}\sum_{\boldsymbol{x}_i \in N_k(x)} \mathrm{RD}(\boldsymbol{x}, \boldsymbol{x}_i)} \tag{4.9}$$

where $N_k(\boldsymbol{x})$ is the set of k-nearest neighbors of point $\boldsymbol{x}$, and $\mathrm{RD}(\boldsymbol{x}, \boldsymbol{x}_i)$ is the Reachability Distance between $\boldsymbol{x}$ and its neighbor $\boldsymbol{x}_i$:

$$\mathrm{RD}(\boldsymbol{x}, \boldsymbol{x}_i) = \max(k\text{-distance}(\boldsymbol{x}_i), \mathrm{distance}(\boldsymbol{x}, \boldsymbol{x}_i)) \tag{4.10}$$

Here, the k-distance$(\boldsymbol{x}_i)$ is the distance to the k-th nearest neighbor of $\boldsymbol{x}_i$, ensuring balanced density estimation across regions with varying point concentrations. The LRD captures local density by considering both the proximity and density of neighboring points.

The Mean LRD (MLRD) for a cluster c_i is computed as the average LRD of all points in the cluster:

$$\mathrm{MLRD}(c_i) = \frac{1}{|c_i|} \sum_{\{\boldsymbol{x} \in c_i\}} \mathrm{LRD}_k(\boldsymbol{x}) \tag{4.11}$$

This provides a measure of cluster cohesion, with higher MLRD values indicating denser and more coherent clusters.

To account for cluster size variations, a scaling mechanism adjusts the cluster score c_i using a Leaky ReLU-like function. The cluster score is defined as:

$$\mathrm{score}(c_i) = \begin{cases} 0 & \text{if} \quad |c_i| < 3 \\ \alpha \cdot f(|c_i|) \cdot \mathrm{MLRD}(c_i) & \text{if } 3 \leq |c_i| \leq E \\ \beta \cdot g(|c_i|) \cdot \mathrm{MLRD}(c_i) & \text{if} \quad |c_i| > E \end{cases} \tag{4.12}$$

where $|c_i|$ is the cluster size, E is the expected cluster size, and $f(|c_i|)$ and $g(|c_i|)$ are scaling functions that penalize small and large clusters, respectively. The minimum threshold of three points is based on PS-InSAR principles, which require at least three spatially coherent PS points to identify deformation processes [Sol+19] reliably.

The final score is computed as the mean across all clusters:

$$\text{Density Score} = \frac{1}{|C|} \sum_{i=1}^{|C|} \text{score}(c_i) \tag{4.13}$$

This comprehensive formulation effectively captures both the local density characteristics and the overall structural properties of ground deformation patterns, while appropriately handling the varying spatial scales and intensities typical of displacement phenomena. In our implementation, the scaling functions $f(|c_i|)$ and $g(|c_i|)$ are defined as $f(|c_i|) = g(|c_i|) = \frac{|c_i|}{N}$, where $|c_i|$ is the cluster size and N is the total number of data points. The expected cluster size E is set to $\frac{N}{k}$ for approximately uniform cluster distributions. Parameters $\alpha = 0.01$ and $\beta = 1$ were calibrated to heavily penalize very small clusters while imposing a lower penalty on larger clusters. The k-nearest neighbors parameter was fixed at $k = 0.98$, ensuring high point density within clusters. These settings optimize the detection of coherent deformation patterns while maintaining balanced cluster distributions.

4.2.5 Baseline Displacement Pattern

To establish reference displacement patterns for validation, we implemented a velocity-based classification framework. This approach segments PS measurements into distinct kinematic categories using threshold-based analysis of displacement time series. Specifically, velocity thresholds (-2 to 2 mm/year) distinguish stable from unstable conditions, considering both individual and combined displacement components. The framework analyzes three dimensions: horizontal displacement (east-west), vertical displacement (up-down), and the integration of these components into movement typologies. For each category, we extract statistical descriptors of temporal evolution, including variability bounds (10th and 90th percentiles) and representative centroids that characterize mean displacement behavior. These descriptors quantify key kinematic parameters such as displacement rates, linearity, and acceleration or deceleration phases.

By integrating horizontal and vertical components, we derive a comprehensive spatiotemporal displacement typology as a physically interpretable reference framework. This baseline provides an objective foundation for assessing clustering performance, as it reflects actual measured displacement directions without relying on external assumptions. Deriving reference patterns directly from velocity measurements avoids subjectivity and lets the data reveal inherent patterns. The methodology aligns with our unsupervised clustering approach, offering a consistent and physically meaningful interpretation of displacement phenomena based solely on measured kinematic properties.

4.3 Results

4.3.1 Optimized Clustering Solution

The application of the online approach (Section 4.2.1) to the total displacement time series identified an optimal number of 8 clusters based on the set parameters (Table 4.3). The total displacement time series captured the complete motion behavior, revealing interaction effects that component-wise analysis would inherently miss. When displacement components are analyzed separately, cluster configurations diverge across different directional measurements, since regions exhibiting uniform behavior in one component can manifest as heterogeneous clusters in another component [HB08]. Such directional discrepancies create inconsistencies in identifying the real number of clusters in the analyzed areas, as the behavior of the ground is determined by the combined effect of all directions of displacement [Zhu+22], [AR23]. Figure 4.6 illustrates the algorithm's progression, where each iteration evaluates the inertia decrease against the threshold criterion. Starting from the domain-driven minimum (gray dashed line), the algorithm detected the most significant improvement in cluster cohesion at $k = 8$ (red line), where the observed inertia decrease exceeded the expected value. Beyond this point, the inertia reduction becomes minimal, and additional clusters fail to provide meaningful improvements in separation, justifying the selection of 8 clusters.

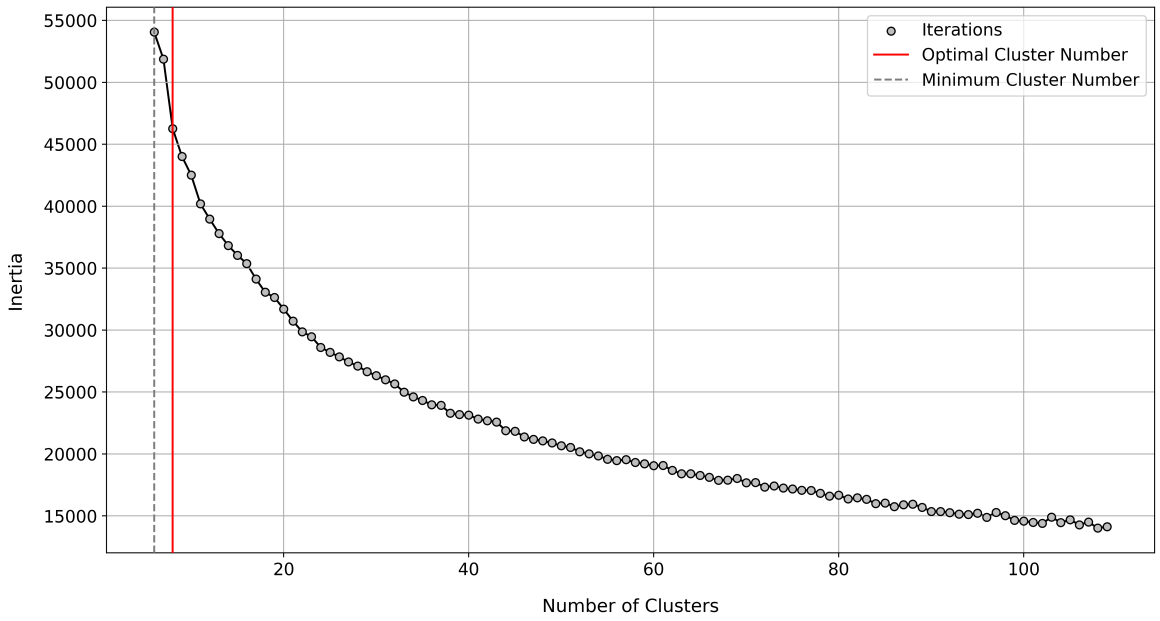


Figure 4.6: Visualization of the inertia curve as a function of the number of clusters, illustrating the application of the optimization approach for determining the number of clusters. The minimum number of clusters is highlighted with a gray dashed line, while the red line marks the optimal value at 8 clusters.

4.3.2 Features Selection

The feature selection process addressed both geospatial attributes and temporal characteristics of displacement patterns. For geospatial attributes, correlation analysis using Pearson and Spearman coefficients identified a strong correlation (0.99) between slope and TRI (Figure 4.7).

Given the focus on slope-driven deformation, the slope was retained as it directly represents terrain gradient, while TRI was excluded. All other parameters (DEM, aspect, TWI, curvatures, and geographic coordinates) showed correlation coefficients below the threshold, confirming their independence and suitability for clustering analysis.

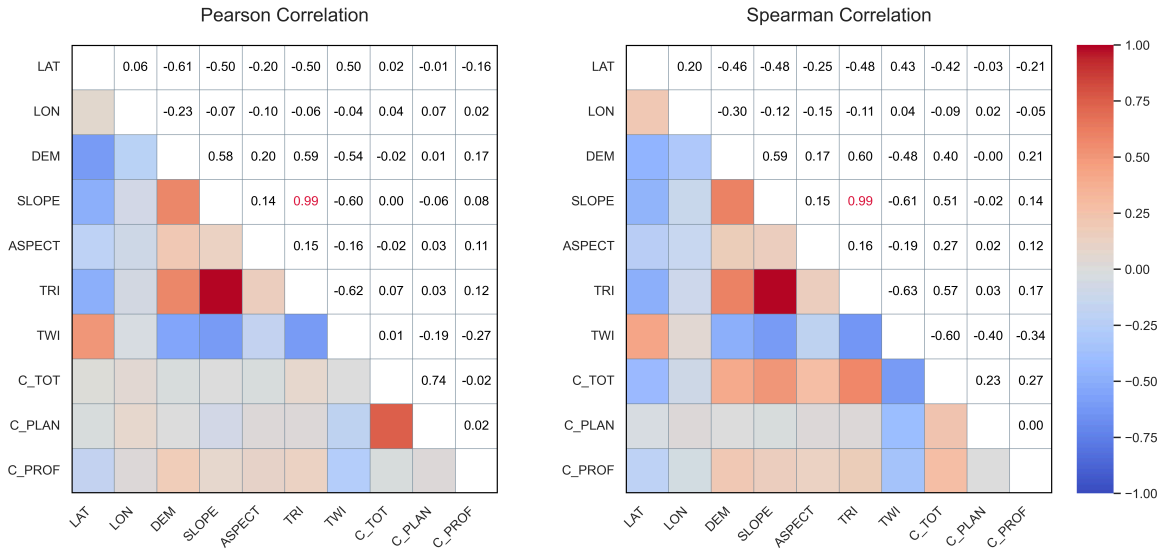


Figure 4.7: Correlation analysis of geospatial attributes: latitude (LAT), longitude (LON), DEM, slope, aspect, terrain ruggedness index (TRI), topographic wetness index (TWI), total curvature (C_TOT}), planar curvature (C_PLAN), and profile curvature (C_PROF). (left) Pearson correlation matrix and (right) Spearman correlation matrix. Attributes with correlation coefficients exceeding the threshold of 0.8 are highlighted in red.

For temporal characteristics, Principal Feature Analysis (PFA) was applied to extract and select features from time series. Table 4.4 provides examples of extracted features, including statistical metrics (e.g., mean change, maximum, entropy) and temporal complexity features (e.g., autocorrelation, first minimum location) for each movement component. Inter-signal features, such as Euclidean and Chebyshev distances, captured relationships between horizontal (EW) and vertical (UD) displacements.

	Intra-Signal						**Inter-Signal**	
	EW			**UD**			**EW+UD**	
	mean change	maximum	entropy	auto-correlation	median	first minimum	Euclidean	Chebyshev
MTS_1	0.004	3.08	1.95	0.79	0.06	0.88	20.71	17.04
MTS_2	0.003	3.72	1.84	0.64	−0.06	0.39	21.99	20.58
MTS_3	−0.005	3.58	1.71	0.46	0.08	0.58	22.48	18.73
⋮	...	...	...	...	...	...	...	...

Table 4.4: Example of interpretable features extracted for the time series analysis. For each MTS, different statistical and signal processing metrics are computed to capture both individual and combined displacement patterns.

From the initial 1742 features, variance-based selection identified the most influential ones at different thresholds. Table 4.5 shows the progressive dimensionality reduction, maintaining essential information while significantly reducing the feature space.

Explained Variance	Number of Features	Feature Types
90%	498	215 intra-signal (EW), 233 intra-signal (UD), 50 inter-signal
80%	310	140 intra-signal (EW), 146 intra-signal (UD), 32 inter-signal
70%	209	92 intra-signal (EW), 93 intra-signal (UD), 24 inter-signal
60%	133	58 intra-signal (EW), 56 intra-signal (UD), 19 inter-signal
50%	74	31 intra-signal (EW), 32 intra-signal (UD), 11 inter-signal
40%	37	13 intra-signal (EW), 16 intra-signal (UD), 8 inter-signal
30%	14	5 intra-signal (EW), 6 intra-signal (UD), 3 inter-signal
20%	9	3 intra-signal (EW), 4 intra-signal (UD), 2 inter-signal
10%	7	2 intra-signal (EW), 3 intra-signal (UD), 2 inter-signal

Table 4.5: Number of features retained at different thresholds. The feature types column indicates the distribution between intra-signal features for each component and inter-signal features.

4.3.3 Clustering Quality Assessment

The analysis of feature-based clustering methods revealed distinct performance patterns across explained variance thresholds (Figure 4.8).

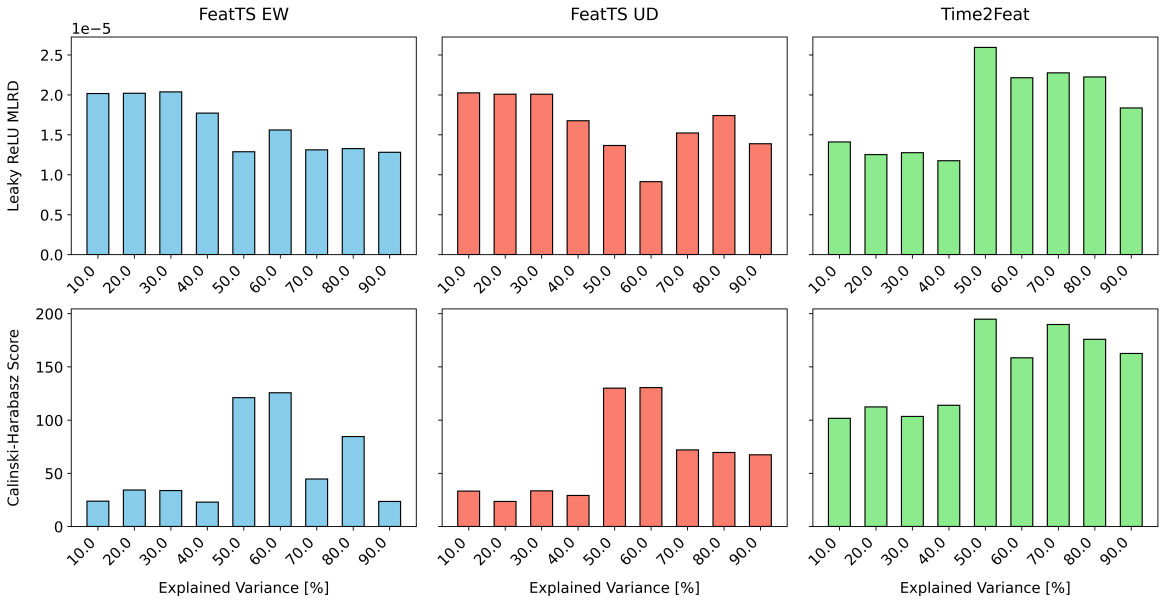


Figure 4.8: Comparison of clustering performance metrics across feature-based methods for varying levels of explainable variance.

For the FeatTS method, the optimal threshold varied distinctly across components and metrics.

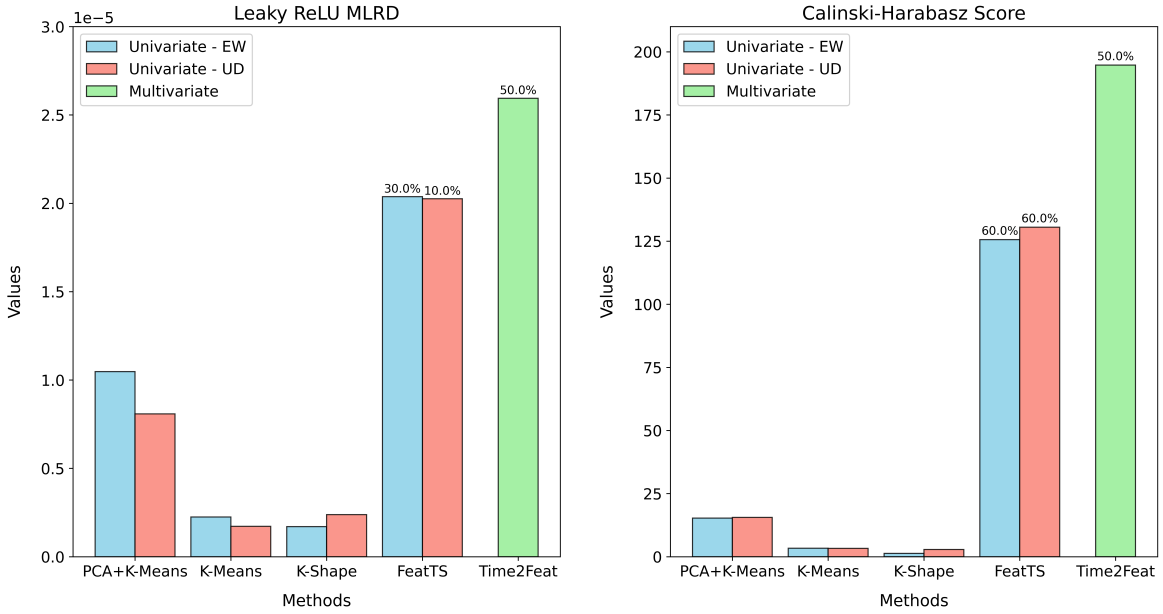


Figure 4.9: Comparison of clustering performance. Higher values indicate better clustering quality. For feature-based methods, only the best-performing results from the threshold analysis are shown. The corresponding explained variance is displayed above bars.

The MLRD values for both components demonstrated peak performance in the lower variance range (10-30%), with the east-west component reaching its maximum of $2.04 \cdot 10^{-5}$ at 30% explained variance and the up-down component achieving $2.03 \cdot 10^{-5}$ at 10(%). Beyond these peaks, both components showed a general declining trend with some fluctuations at higher thresholds (60% for east-west and between 70-80% for up-down). The CH score exhibited markedly different behavior across the variance spectrum. Both components maintained relatively low values (approximately 20-35) up to 40% explained variance, followed by a sharp increase at

60%. At this threshold, the east-west component reached 125.61 and the up-down component peaked at 130.52, after which both showed a substantial decrease. The multivariate Time2Feat approach demonstrated a distinct pattern of moderate performance until 40% explained variance, followed by optimal results at 50% for both metrics (MLRD: $2.59 \cdot 10^{-5}$, CH: 194.73). Beyond this peak, performance gradually decreased while maintaining values higher than those observed in the 10-40% range.

After identifying the optimal variance thresholds, clustering performances were evaluated across all approaches (Figure 4.9). Raw-based methodologies showed lower performance across both indexes. K-Means achieved MLRD values of $2.25 \cdot 10^{-6}$ for east-west and $1.72 \cdot 10^{-6}$ for up-down components, while K-Shape demonstrated similar performance with $1.70 \cdot 10^{-6}$ and $2.38 \cdot 10^{-6}$ respectively. The CH scores for these methods remained consistently low, ranging from 1.28 to 3.34. The integration of PCA preprocessing with K-Means clustering demonstrated notable improvements over standard K-Means, achieving intermediate MLRD values of $1.05 \cdot 10^{-5}$ for east-west and $8.08 \cdot 10^{-6}$ for up-down displacements, with CH scores of approximately 15 for both components.

Feature-based approaches demonstrated higher clustering quality. At their optimal thresholds, both FeatTS components significantly outperformed raw-based methods, highlighting the enhanced cluster separation achieved through PFA feature extraction. The multivariate Time2Feat method exhibited the highest overall performance among all approaches, representing improvements of approximately one order of magnitude in MLRD and two in CH score.

The first screening of displacement velocities reveals distinct patterns across the study area. The univariate analysis (Figure 4.10) shows that $\sim 84\%$ of the study area maintained stable conditions in both horizontal and vertical directions.

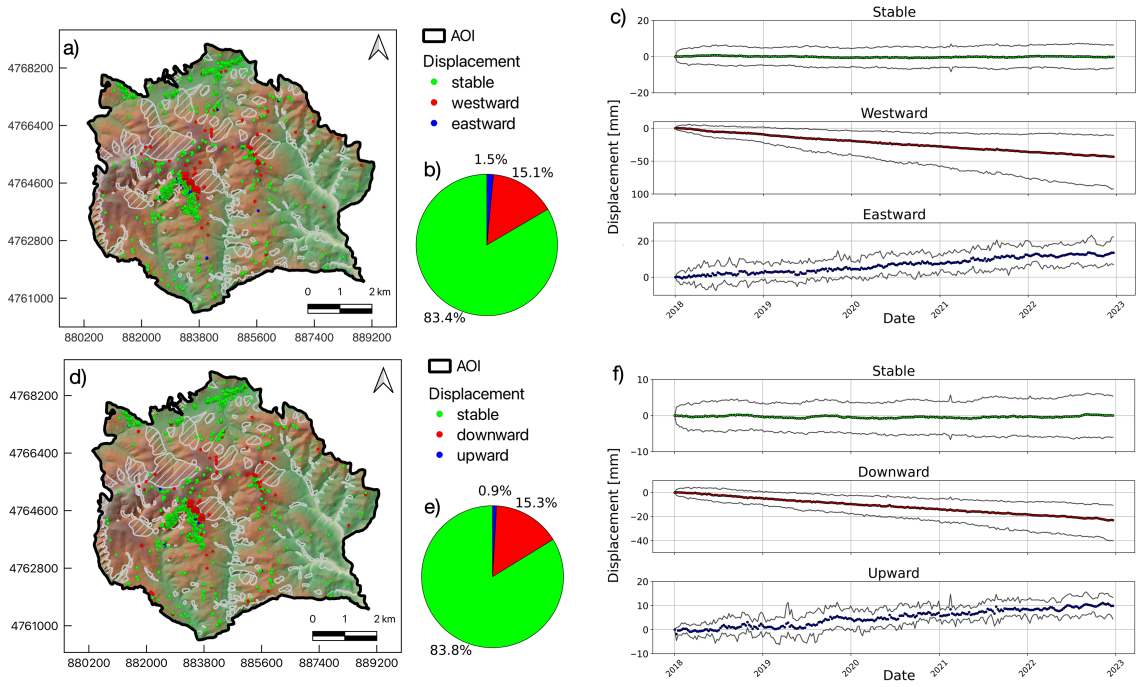


Figure 4.10: Univariate analysis of ground displacement. (a,d) Spatial distribution and temporal evolution of horizontal (top) and vertical (bottom) displacement components. (b,e) Pie charts indicate the percentage distribution of movement types. (c,f) Displacement trends for each movement category, with 10th, and 90th percentiles (black series) and the centroids.

In the east-west direction, 15.1% of the area exhibited westward movement with cumulative displacements reaching up to −100 mm, while the 1.5% showed eastward movement with gradual increases of up to 20 mm.

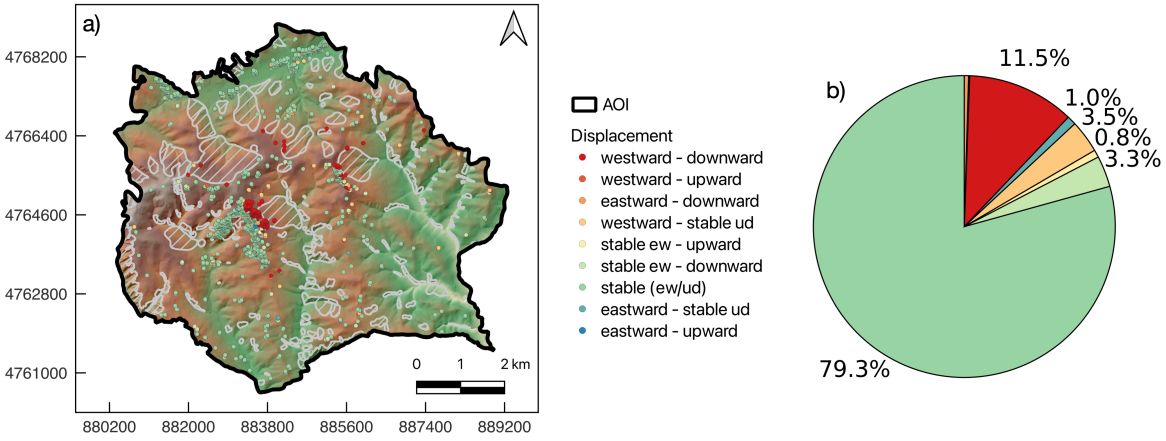


Figure 4.11: Multivariate analysis of ground displacement. (a) Spatial distribution of combined movement types across the study area. (b) The pie chart illustrates the percentage distribution of different movement combinations. The movement categories are classified based on the combination of east-west and up-down displacement components.

Similarly, in the vertical direction, 15.3% of the area experienced downward movement with maximum displacements of approximately −40 mm, and 0.9% displayed upward movement of up to 10 mm. The temporal evolution of horizontal and vertical displacements demonstrate similar temporal patterns, with centroids of westward and downward movements showing a steady linear rate, while eastward and upward movements display more irregular patterns with periods of acceleration and deceleration.

An initial MTS analysis (Figure 4.11) combines these directional classifications, indicating that 79.3% of the area maintained overall stability in both directions. The westward-downward movement emerged as the predominant combined displacement pattern, affecting 11.5% of the area. Other significant combined movements include eastward-upward displacement at 3.5% and stable east-west with downward movement at 3.3%. The remaining displacement combinations each affected less than 1% of the total area. This initial spatial assessment shows a clear relationship between displacement patterns and topography, with stable areas predominantly located in regions with lower elevations and gentler slopes. The most significant combined displacement (westward-downward), typically within or along buffer zones of mapped landslide perimeters, is predominantly concentrated in the central-western portion of the study area. Transition zones between stable and unstable regions often combine stable and in-displacement patterns along different components.

4.3.4 Local-Scale Analysis: Offida Landslide

A local-scale analysis of the multivariate clustering results was focused on the eastern portion of the Offida urban area (Figure 4.12), affected by a landslide with an extension of approximately 0.5 km^2, classified as high-risk in the PAI inventory.

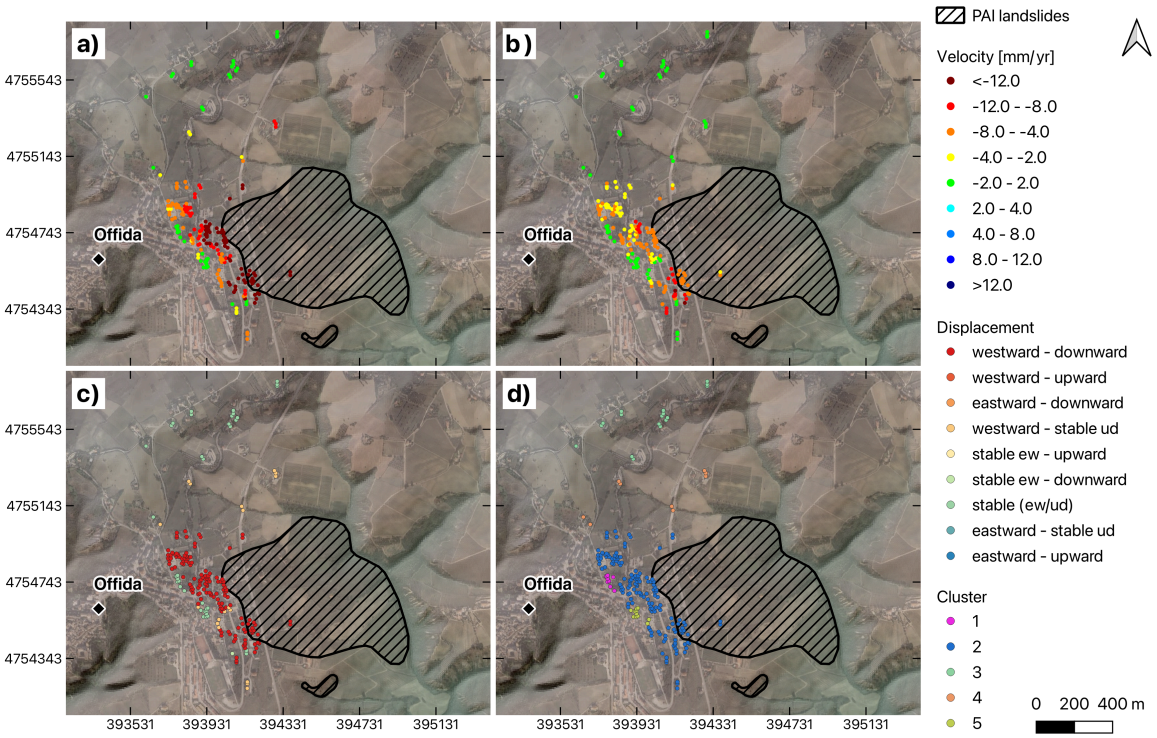


Figure 4.12: Spatial analysis of displacement patterns in the Offida landslide area. a) East-west and b) up-down velocity distributions. c) Combined displacement showing the interaction between horizontal and vertical components. d) Results of Time2Feat, MTS clustering.

The investigated area exhibits complementary deformation patterns in horizontal and vertical velocities (Figure 4.12 a, b). The western portion of the landslide shows notable displacement reaching velocities up to 25 mm/yr. Combined analysis of displacement components (Figure 4.12 c) reveals a dominant westward-downward movement pattern, particularly concentrated along and beyond the mapped boundary. Time2Feat analysis identified well-separated clusters with distinct spatiotemporal characteristics (Figure Figure 4.12 d). Cluster 2 exhibits deformation patterns distributed both within and beyond the mapped landslide boundaries, while Clusters 1 and 5 show different displacement characteristics, despite their spatial proximity to the landslide area. Cluster 3 occupies stable portions of the study area, while Cluster 4 represents the transition zone between stable and unstable areas. The spatial distribution of Cluster 2 extends significantly beyond the current PAI perimeter, suggesting that the main landslide movement affects a substantially larger area than originally mapped. Comparing Time2Feat clustering results with the baseline displacement patterns confirms the effectiveness of the multivariate approach in identifying coherent deformation zones. The clustering successfully differentiates between areas with similar temporal evolution characteristics, providing richer information than velocity-based classification alone. The alignment between Cluster 2 and the westward-downward movement pattern validates the clustering's ability to capture physically meaningful deformation processes.

The distinct clustering patterns enable domain experts to interpret complex landslide behavior with greater precision, complementing previous studies [Fra+18], [Cri+21] on ground displacement. By revealing functional components within landslide systems, from active deformation zones to transitional areas, experts gain insight into the underlying kinematic behavior and can develop targeted mitigation strategies based on the specific characteristics of ground deformation.

This approach is scalable for integration into large-scale assessment frameworks as proposed by [Sol+19], [Zoc+25], or for temporal characterization of trigger-response relationships in ground deformation processes as proposed by [Bia+18], [Gha+24a]. In both applications, the method provides consistent evaluation while maintaining the site-specific interpretability necessary for effective risk management across diverse geographic settings.

4.3.5 Temporal Robustness Analysis

To evaluate the temporal robustness of our methodology, we validated it by dividing the time series into temporal subsets. We analyzed the first half of the monitoring period (ending April 21, 2020) separately from the full 5-year dataset. Figure 4.13 shows the clustering results for the half-period analysis of the Offida landslide area, comparable with the full-period results in Figure 4.12. The comparison reveals identical clustering patterns, with consistent spatial distributions and relationships to the mapped landslide boundaries. This consistency is supported by the analysis of displacement trends in the area, which confirmed predominantly linear time series behavior throughout the monitoring period, without significant trend changes that would influence the clustering solution. The results indicate that the feature-based multivariate approach effectively identifies stable deformation signatures that persist across different temporal windows. While the findings do not rule out temporal variations in the landslide's physical behavior, they demonstrate that the key characteristics captured by our feature extraction remain dominant. The clustering stability demonstrates that our methodology reliably isolates significant spatiotemporal patterns, ensuring consistent identification of ground deformation zones across varying time periods.

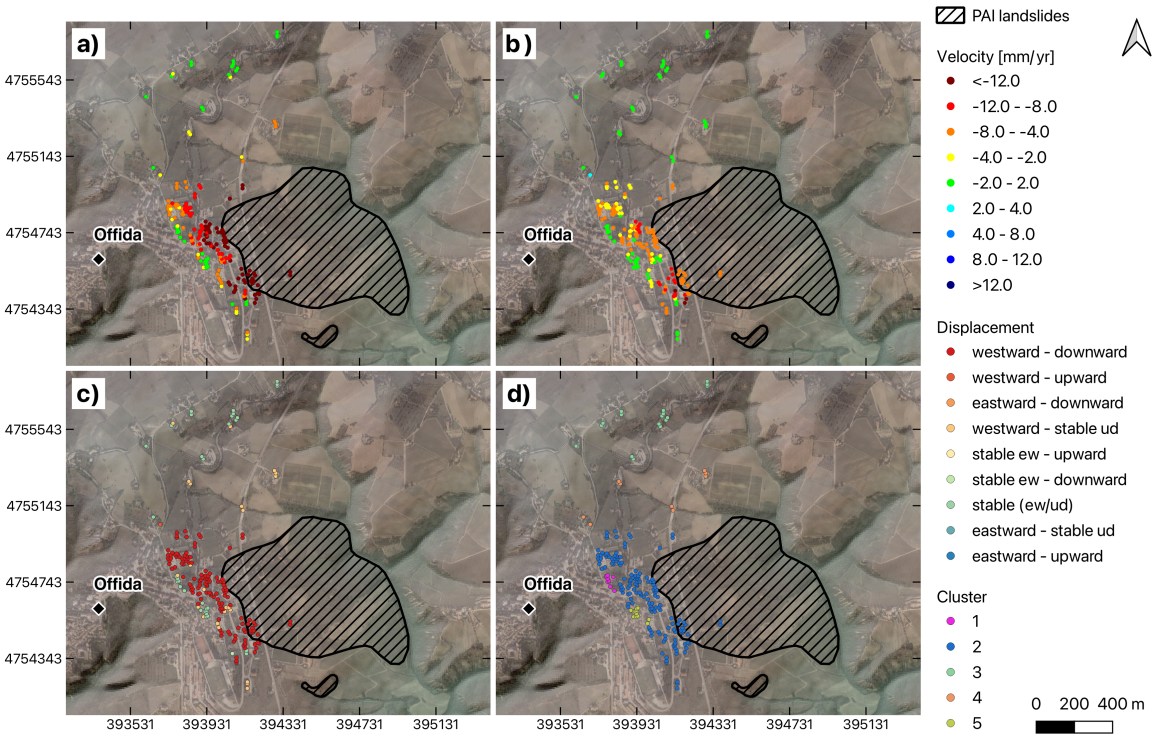


Figure 4.13: Temporal robustness analysis of the Offida landslide area using the first half of the monitoring period (ending April 21, 2020). a) East-west and b) up-down velocity distributions. c) Combined displacement showing the interaction between horizontal and vertical components. d) Results of Time2Feat, MTS clustering.

4.4 Discussion

The characterization of displacement patterns through PS data analysis reveals distinct performance differences across clustering approaches. Feature-based clustering methods outperform raw-data approaches by one to two orders of magnitude in both MLRD and CH scores. This performance gap demonstrates that standard clustering of displacement time series fails to capture the complexity of deformation patterns, underscoring the value of feature extraction. These findings align with studies by [Bia+18], [Wan+24c], who demonstrated improvements when applying multi-temporal feature extraction approaches to landslide monitoring. While PCA-based dimensionality reduction provides some improvement, PFA-based methods achieve superior results by extracting physically meaningful features that better represent ground deformation processes.

Analysis of explained variance thresholds reveals optimal ranges for information retention in feature-based approaches. The effectiveness of relatively low thresholds (50-60%) indicates that significant dimensionality reduction can be achieved while preserving essential pattern information. This finding has practical implications for processing large PS datasets, enabling efficient analysis without compromising detection capabilities.

The comparative analysis between univariate and multivariate approaches provides insights into component interactions in ground deformation processes. While FeatTS achieves notable improvements over raw-based methods, its optimal performance requires different explained variance thresholds for MLRD (30% for east-west, 10% for up-down) and CH scores (60% for

both components). This discrepancy suggests that different aspects of the deformation signal are captured at different levels of feature compression. The multivariate Time2Feat approach demonstrates superior performance over univariate analysis with consistent optimal thresholds (50% explained variance) for both metrics, suggesting a more robust and unified characterization of the deformation patterns.

This multivariate advantage aligns with the physical nature of ground deformations, where horizontal and vertical movements are mechanically coupled. [Guo+06], [SKS20], [Rig+23] established that multivariate analysis significantly enhances the detection of complex geospatial patterns in displacement data, providing statistically more reliable results compared to univariate approaches through comprehensive capture of directional interactions. The ability to simultaneously analyze multiple displacement components enables the detection of subtle pattern variations that might be missed in separate directional analyses, particularly in transition zones between stable and unstable areas where component interactions can reveal early signs of progressive deformation.

Moreover, multivariate techniques such as Independent Component Analysis (ICA) [Fer+20], [Bel+97] enhance signal clarity by separating geophysically meaningful deformation patterns from atmospheric artifacts and orbital errors. This separation is crucial in large-scale monitoring systems, such as PS InSAR time series, where mixed-source signals can obscure early deformation signatures. By isolating independent sources, multivariate methods improve detection sensitivity and enable automated, unsupervised interpretation in data-rich environments [Gad+18]. Additionally, multivariate analysis proves highly effective for identifying outliers in PS-InSAR datasets, where large observation volumes and varying noise conditions challenge classical filtering techniques. By jointly considering multiple parameters (e.g., displacement, coherence, DEM height) and spatial relations among scatterers, multivariate deep learning models such as Convolutional Neural Networks (CNNs) can robustly distinguish true deformation signals from anomalous or low-quality observations [Agu+21]. This facilitates early hazard detection and improves the overall reliability of geospatial monitoring frameworks. Further validation of multivariate clustering comes from [Rig+23], who demonstrated its effectiveness in classifying complex ground deformation phenomena on local scales. Their work confirmed that multivariate approaches successfully identified coherent deformation clusters, outperforming univariate methods in separating overlapping deformation signals. These validation results support our methodological framework, confirming that feature-based multivariate clustering provides a more thorough characterization of ground deformation patterns.

Application to the Offida landslide builds on these validated methodological approaches, revealing deformation patterns beyond previously mapped boundaries. The multivariate approach's superior clustering quality provides clear discrimination between movement zones with distinct kinematic behaviors, enabling more precise identification of unstable areas and their spatial extent. Furthermore, as demonstrated through the temporal subset analysis, the methodology's ability to consistently identify these patterns across different time periods ensures reliable long-term monitoring capabilities. This enhanced delineation enables engineers and geologists to precisely identify unstable areas and understand their spatial relationships, supporting more targeted field investigations and monitoring strategies. The practical impact of this detection capability extends directly to hazard assessment and infrastructure management. By providing quantitative evidence for updating existing inventories and improving understanding of ongoing deformation processes, the approach bridges the gap between large-scale remote sensing analysis and site-specific targeted interventions for actionable decisions in land-use planning and management.

Additionally, by preserving the original physical parameters throughout the analysis, the feature-based approach ensures transparent and traceable results that domain experts can directly validate and integrate with conventional geological surveys. This interpretability becomes especially valuable when the findings challenge existing knowledge, as demonstrated in the case study of the Offida landslide. The methodology's transferability makes it applicable across diverse landscapes and geological settings, allowing its integration into broader risk assessment frameworks [CBC13], [Cri+21], [Zoc+25] or detailed characterization analyses of ground deformation processes [CA21], [Zhu+22], [Gha+24b].

The robustness of the proposed methodology was validated through multiple approaches. Temporal stability analysis (Section 4.3.5) showed consistent clustering patterns across different time windows, with identical spatial distributions and boundary relationships observed for both the full 5-year period and the first half. Sensitivity analysis of the threshold parameter θ in the online optimization algorithm confirmed stable results (8 clusters) for values above 0.1, indicating that the clustering reflects inherent data patterns rather than artifacts. Comparative analysis with the baseline displacement pattern (Section 4.2.5}) further validated the approach, as Time2Feat clustering showed strong alignment with multivariate displacement typologies derived from velocity classifications.

4.4.1 Novelty, Limitations, and Future Developments

The findings of this study align with previous research highlighting the complexity of ground deformations and the need for advanced characterization techniques [ZZZ22], [Jin+23], [Fes+23], [Pen+24], [Zoc+25]. This research introduces several methodological developments, including the MLRD score, specifically designed to evaluate PS clustering quality. Unlike traditional metrics, MLRD effectively handles irregular spatial distributions and varying densities by combining local density estimation with relative distance measures, providing a robust evaluation framework aligned with the physical nature of deformation processes.

The online optimization approach for determining cluster numbers improves upon previous PS-InSAR methods. While studies like [Fes+23] used PCA with K-means and [MS11] applied the CLARA algorithm, these methods lack integration with domain-specific constraints. Others, such as [Bia+18], avoid predefining cluster numbers, instead identifying anomalous points based on spatial and temporal consistency. Our approach dynamically evaluates inertia reductions against adaptive thresholds while incorporating minimum cluster constraints informed by geomorphological knowledge. This integration of statistical optimization with physical insights enables automatic, objective determination of cluster numbers, yielding configurations that better represent complex deformation mechanisms.

The proposed feature-based framework advances beyond previous studies relying on PCA-derived components [Fes+23] or raw time series [ZZZ22], [Zoc+25]. It preserves interpretability by extracting and selecting physically meaningful features, such as statistical metrics and signal processing parameters, directly from time series. Dimensionality reduction through explainable variance thresholds (50-60%) achieves an optimal balance between analytical rigor and feature complexity. Additionally, integrating geospatial attributes enables a comprehensive spatiotemporal analysis of deformation mechanisms, advancing beyond purely temporal approaches [Jin+23], [Pen+24]. Multivariate clustering further enhances insights by simultaneously analyzing horizontal and vertical displacement components.

While providing significant advantages, the methodology has some limitations. Current SAR satellite geometry restricts north-south displacement detection, limiting the analysis to two-

dimensional fields. This constraint will be addressed by future satellites with highly inclined orbits capable of retrieving north-south displacements. Additionally, applying the approach to medium- and large-scale analyses faces challenges due to time series variability across different territories, potentially affecting clustering consistency. Partitioning the analysis into morphologically homogeneous units and integrating results could address these challenges, maintaining accuracy across broader scales.

Future developments could also integrate semi-supervised learning to enhance feature-based clustering by incorporating expert knowledge through labeled data. This approach is particularly promising for regions with sparse or uncertain data or areas with high variability or strong gradients. However, balancing expert supervision with analytical objectivity remains a key challenge requiring further investigation.

4.5 Conclusions, Contributions and Future Work

This study presents a comprehensive data-driven framework for characterizing ground deformations through PS-InSAR analysis, addressing the challenges of complex spatiotemporal patterns in large-volume displacement measurements. The methodology advances traditional clustering approaches by integrating feature extraction techniques with multivariate analysis, enabling thorough characterization of displacement patterns and their intrinsic relationships.

Our research introduces several methodological innovations including i) an online optimization approach for automatic cluster number determination, ii) feature-based clustering methods that preserve interpretability, iii) the integration of geospatial attributes for enhanced pattern recognition, iv) multivariate time series analysis implementation, and v) the formulation of density-based score for PS data clustering evaluation. These advancements collectively provide a robust framework for analyzing ground deformation processes.

Results demonstrate feature-based clustering's superiority over raw-data approaches, with one to two orders of magnitude improvement in quality metrics. The multivariate Time2Feat approach achieved optimal performance (MLRD: $2.59 \cdot 10^{-5}$; CH: 194.73) at 50% explained variance. PFA reduced features from 1742 to 74 while preserving explainability and enhancing clustering quality. This dimensionality reduction improved computational efficiency with meaningful pattern extraction. Applied to the Offida municipality, our methodology identified coherent deformation clusters beyond mapped landslide boundaries, with westward-downward movements affecting 11.5% of the area. Temporal validation confirmed robust clustering patterns across different timeframes. Local-scale analysis showed Cluster 2 extending beyond PAI boundaries, demonstrating effectiveness in detecting unmapped unstable areas.

The implications of this research extend significantly into practical geohazard assessment and risk management. The interpretable and multi-dimensional analysis supports effective hazard monitoring and improved characterization of ground deformation variability. This comprehensive approach provides quantitative support for updating hazard maps and understanding ongoing deformation processes, directly benefiting urban planning and risk assessment in susceptible areas.

This work was conducted together with Claudia Masciulli, Donato Tiano, Marta Zocchi, Francesco Guerra, Paolo Mazzanti, and Gabriele Scarascia Mugnozza [Mas+25]. The capability to approach this problem merging geological knowledge (C. Masciulli, M. Zocchi, P. Mazzanti, and G. Scarascia Mugnozza) with I.T. data-driven approaches (G. Guiduzzi, D. Tiano, and F. Guerra) allowed the definition of a pipeline that, taken a PS-InSAR dataset describing the

mobility of points in time, is able to apply different clustering techniques and evaluate their performance through the proposed MLRD metric, which integrates point density assessment with feature-based similarity measures. Through this pipeline, the team was able to gather sufficient data to prove that feature extraction is an effective method to address the limitations of current PS-InSAR clustering techniques.

Future research directions include extending multivariate analysis across diverse geological contexts and larger spatial scales to advance our understanding of complex ground deformation processes. The potential integration of additional data sources and exploration of semi-supervised learning techniques could further enhance the methodology's effectiveness in characterizing ground instability phenomena, particularly in areas with diverse environmental settings and complex deformation patterns.

4.5.1 Discussion on Methodological Generality and Geographic Specificity

While the pipeline developed in this work — encompassing feature extraction, dimensionality reduction, and MLRD-based cluster evaluation — is designed as a general-purpose spatio-temporal discovery framework, its current instantiation is grounded in the specific geological and urban context of Offida, Italy.

The performance of the clustering was influenced by the high density of Persistent Scatterers (PS) available in the urbanized areas of Offida and the specific "Up-Down" and "East-West" deformation patterns characteristic of its landslide-prone clayey-sandy lithology. Results observed here, such as the optimal performance of certain feature subsets, may be tied to the specific signal-to-noise ratio and the revisit time of the satellite orbits (ascending and descending) used for this dataset.

To claim universal applicability across diverse geologies (e.g., volcanic regions, permafrost areas, or coastal subsidence zones), the methodology would require:

1. **Cross-Geological Validation**: Testing the MLRD metric on regions with significantly different deformation kinematics (e.g., non-linear or purely vertical motion).
2. **Sensor Agnostic Testing**: Evaluating the feature extraction layer on data from different satellite constellations (e.g., Sentinel-1 vs. COSMO-SkyMed) with varying spatial resolutions.
3. **Scalability Stress-Tests**: Assessing the computational performance of the clustering pipeline on regional-scale datasets (millions of PS) to verify that the spatio-temporal discovery remains efficient beyond the localized urban scale.

CHAPTER 5

Foundations of Compact Data Structures for Efficient Time Series Storage

The exponential growth of TSD across industrial IoT, financial markets, and environmental monitoring has necessitated a paradigm shift in storage architecture. Traditional compression methodologies, such as those implemented in the Lempel-Ziv family (e.g., GZIP) or block-based columnar formats, prioritize the maximization of compression ratios at the expense of data accessibility. While effective for archival purposes, these "black-box" approaches require the full decompression of a data block to retrieve even a single temporal point, introducing significant latency in high-frequency analytical workflows and real-time dashboarding. To address this bottleneck, the field of **C**ompact **D**ata **S**tructures (CDS) has emerged as a critical research frontier. The core objective of CDS is to represent information in a space-efficient manner while supporting direct query operations — such as random access, rank, and select — on the compressed representation itself without necessitating a prior decompression stage. In the context of time series, this implies an infrastructure where statistical summaries, motif searches, and range queries can be executed directly within the compressed domain, effectively merging the storage and indexing layers.

A fundamental challenge in applying CDS to TS lies in the tension between construction time complexity and reconstruction efficiency. The "Efficiency Pillar" of this thesis posits that for a compression scheme to be viable in modern temporal infrastructures, it must not only provide high bit-savings but also facilitate rapid construction that can keep pace with incoming data streams. Among the candidates for such infrastructure, the **Relative Lempel-Ziv** (**RLZ**) algorithm stands out due to its asymmetric nature. RLZ leverages a static "reference" string to identify and encode redundancies via pointers, offering near-instantaneous random access – a property vital for interactive temporal dashboards. However, the efficacy of RLZ is intrinsically bound to the quality of its reference. Historically, this reference has been constructed through stochastic sampling or fixed-length chunking, methods that often fail to capture the multi-scale repetitive patterns (**motifs**) inherent in complex temporal signals.

This chapter investigates the transition from heuristic-based motif extraction to formal algorithmic approaches. In this research, an architectural system was developed to replace an arbitrary reference with a "data-aware" one, constructed through suffix structures and optimized via a length-frequency scoring metric. This explores how RLZ can be adapted from a high-speed stochastic compressor into a queryable architecture for temporal repositories.

5.1 Relative Lempel-Ziv (RLZ) and the Reference Problem

The Relative Lempel-Ziv (RLZ) algorithm is a dictionary-based compression scheme specifically engineered for high-performance random access on repetitive data. Unlike traditional LZ77 [LZ77], which references previously occurring patterns within the same text, RLZ factorizes a target text T by referencing a separate, static reference string R. Formally, RLZ factorizes T into z phrases $f_1, f_2, ..., f_z$ such that $T = f_1 \cdot f_2 ... f_z$. Each phrase f_i is defined as the longest prefix of the remaining suffix of T that exists as a substring within R. Each phrase is typically encoded as a pair $\langle s_i, l_i \rangle$, where s_i is the starting position (source index) of the match in R, and l_i is the length of the match. The primary advantage of this scheme is constant-time random access; to retrieve $T[x]$, one simply identifies the phrase covering x and directly accesses the corresponding offset in R [Din+25]. The compression efficiency of RLZ is strictly bounded by how effectively R captures the underlying redundancy of T. In early or default implementations, R is often constructed by stochastic sampling, where R is formed by concatenating k random chunks (segments) of size s from T:

$$R = \text{chunk}_1 \cdot \text{chunk}_2 \cdot \ldots \cdot \text{chunk}_k \quad \text{where} \quad \text{chunk}_i \in T \tag{5.1}$$

This "random chunk" approach is mathematically suboptimal for several reasons:

- Random sampling does not prevent the inclusion of the same frequent patterns multiple times within R. If k random chunks contain the same highly frequent motif m, the "entropy" of the reference is wasted, as a single occurrence of m would suffice to compress all instances of m in T.
- A random chunk might start or end in the middle of a frequent motif. If a motif m is split across two chunks, the reference cannot provide a single long match for m, forcing the RLZ parser to break m into multiple smaller, less efficient phrases.
- Let $P(m)$ be the probability of a motif m appearing in T. In a random reference of size $|R|$, the probability that a specific motif m of length $|m|$ is captured in its entirety is approximately:

$$P(\text{capture}) \approx 1 - \left(1 - \frac{|R| - |m|}{n}\right)^k \tag{5.2}$$

As the complexity and number of motifs in T increase, the probability that a random reference captures a complete and diverse set of these motifs diminishes, leading to an increase in the number of phrases z required to represent T.

The core research hypothesis of this pillar is that the random sampling process can be replaced by a deterministic selection of motifs. By identifying subsequences that maximize both length (to reduce the total number of phrases) and frequency (to increase the probability of a match), it is possible to construct a reference that is mathematically "dense" with information. If R is constructed from an optimal set of motifs M, the compression ratio improves as $z \to z_{\min}$, where $z_{\min}$ is the theoretical minimum number of phrases achieved when every phrase in T finds its longest possible match in R. This chapter investigates the architectural viability of leveraging

Suffix Arrays to approximate this optimal set. It explores whether algorithmic string-matching can mitigate the latency of distance-based metrics — acting as a high-speed heuristic to navigate the brute-force search space of variable window sizes — and rigorously evaluates its structural efficacy against the highly resilient stochastic baseline.

5.2 Methodology

5.2.1 Motif Discovery via Matrix Profiles (STUMPY)

The first phase of the architectural investigation explored the use of the Matrix Profile as a mechanism for identifying repetitive subsequences. Utilizing the STUMPY library, the objective was to extract motifs that could serve as the "educated foundation" for the RLZ reference string. The initial implementation relied on computing the Matrix Profile across a vast set of window sizes (m), ranging from seconds to years (e.g., $m \in \{5s, 10s, ..., 10y\}$). While STUMPY provides an efficient implementation of the STOMP and GPU-STUMP algorithms, the computational cost remained a significant architectural constraint:

- For every distinct value of m, a full Matrix Profile must be computed. Even with GPU acceleration, the total time complexity scaled linearly with the number of motif sizes being evaluated ($O(n^2 \cdot |m_{\text{values}}|)$).
- This approach essentially functions as an "efficient brute force", re-scanning the entire time series for every window size without leveraging the overlapping information found in previous iterations.

The procedure to build the reference relies on maintaining an `OrderedSet` of unique data points to ensure temporal coverage. To keep track of the portions of the time series covered by discovered motifs, an `IntervalTree` was adopted, effectively indexing a set of intervals to perform spatial queries (e.g., checking overlaps). The IntervalTree approach allowed for an incremental coverage logic, where intervals could be added as completely new or extended with adjacent sequences. As motifs were discovered, the script calculated the "newly added points" to determine if a motif was worth merging or if it constituted redundant temporal coverage.

To maintain the scalability of the reference construction phase, a non-overlapping motif constraint was enforced. This design choice prevents the pseudo-quadratic re-evaluation cost associated with iterative frequency updates, although evaluating the merge of motifs that overlap could improve the entropy captured by the reference. Empirical observations during the construction phase highlight a computational bottleneck in brute-force motif discovery. While STUMPY ensures optimal motif selection, the $O(N^2)$ complexity introduces a latency that defines the upper bound of the system's scalability for real-time indexing, necessitating a more structured architectural solution for multi-scale contexts.

5.2.2 Suffix Arrays and the LF-Score

To overcome the scalability limitations of the STUMPY-based approach, the architecture was redesigned around a dedicated `ReferenceBuilder` module. This transition shifted the paradigm from distance-based motif discovery to algorithmic string-matching, specifically leveraging **Suffix Arrays** (**SA**) to identify redundancies across all temporal scales simultaneously. By mapping the

time series into a discrete integer alphabet and constructing a Suffix Array via `pydivsufsort`, the system could query the global frequency of any subsequence in $O(m)$ time. Unlike the Matrix Profile, which is bound to a fixed window m, the Suffix Array allows for the evaluation of variable-length motifs without re-computing the entire underlying structure. The builder utilizes binary search over the suffix array to find the exact number of occurrences for any candidate motif, ensuring the reference is populated strictly with genuinely repetitive structures.

The proposed architecture introduces the **LF-Score**, a weighted metric engineered to prioritize subsequences for the RLZ reference. The score defines the mathematical trade-off between a motif's length (L) and its frequency (F) to maximize overall compression power. It is formulated as:

$$\mathrm{LF}(s) = w_l \log(\text{length}) + (1 - w_l) \log(\text{frequency}) \tag{5.3}$$

where w_l (the **Length Priority Factor**, or **LPF**) allows for the prioritization of either longer subsequences (which reduce the total number of RLZ pointers) or more frequent ones (which increase the probability of a match). By normalizing these values into a [0...1] range, the builder can greedily select the most optimal motifs to exhaust the predefined reference budget.

The final reference is constructed through an iterative, greedy deterministic process:

1. All discovered motifs are sorted by their LF-Score in descending order.
2. Motifs are appended to the `Reference` buffer until the predefined `reference_size` boundary (e.g., 10% of the original data) is reached or the set of available motifs is exhausted.
3. The architecture utilizes the `IntervalTree` to systematically manage temporal overlaps, tracking uncovered segments and allowing the builder to structurally salvage partially overlapping motifs if they yield a net positive global LF-Score.

This deterministic, data-aware strategy represents an architectural attempt to transition from the randomness of standard RLZ to a rigorous structural mapping. However, its overall computational viability and compression efficacy must be carefully evaluated against the resilient stochastic baseline.

5.2.3 Implementation and Interoperability: Python-C++ Pipeline

To establish a rigorous benchmark comparing the "educated" reference against the stochastic baseline, a cross-language pipeline was engineered. This architecture prioritizes the decoupling of the motif discovery phase (high-level Python logic) from the compression execution phase (high-performance C++ processing). This structural split guarantees that performance benchmarks are aligned with the state-of-the-art literature regarding RLZ, remaining unbiased by interpreter overhead.

The reference subsequences mathematically identified by the Python module are serialized into a custom binary format. This preserves the precision of the normalized time series while ensuring the data structures are packed efficiently for low-level memory loading. The resulting binary reference is fed into a C++ implementation of the RLZ algorithm. This environment facilitates direct benchmarking between the custom reference (LF-scored motif collection) and the naive

reference (stochastic population via random 1024-byte chunks). The naive C++ baseline was integrated via collaboration with Dr. Adrián Gómez-Brandón (ORCID page).

5.3 Empirical Observations and Complexity Analysis

The empirical investigation into motif-based reference construction yielded critical insights into the physical limits of deterministic optimization for temporal compression. While the Suffix Array and LF-Score methodology successfully provided a data-aware foundation, the empirical results highlight a complex architectural trade-off between motif exhaustion and stochastic coverage.

5.3.1 Architectural Trade-offs and Limitations

A critical scaling bottleneck was observed when the required reference size exceeded 1% of the original data volume. In these boundary conditions, the algorithm effectively "ran out" of statistically significant motifs before the reference budget was fully saturated. Consequently, the remaining reference space defaulted to less frequent or shorter subsequences, triggering a "density dilution" that actively degraded the overall compression power.

To aggressively stress-test this bottleneck, a secondary heuristic approach was deployed where motif sizes were dynamically adapted based on data-chunk sizes (powers of 2) rather than rigid temporal intervals. By querying the RLZ algorithm to identify the "largest factor" found during a naive pass, the builder deterministically replicated the most effective chunk topologies. While this optimized the computational efficiency, the architectural bounds of deterministic extraction remained rigid, proving insufficient to completely resolve the density dilution.

Systematic experimental logs indicate that the Naive RLZ (random reference) maintained a marginal but persistent lead in compression efficiency, consistently outperforming the Custom Reference by 1% to 3%. The custom reference secured compression ratios in the range of 21.7% to 25.5%, scaling alongside the Length Priority Factor (LPF); in contrast, the naive stochastic reference maintained a 1% to 3% lead in compression efficiency, reaching as low as 20.4% under specific configurations.

5.4 Synthesis of Observations: Toward a Validated Architecture

The architectural investigation into Suffix Array-driven compression yields several critical insights into the design space of queryable temporal infrastructure. While this work remains exploratory, the following observations establish the boundaries for future validated implementations:

- The Brute-Force Scaling Limit: The testing of STUMPY for motif extraction indicated that while deterministic discovery provides a more structured approach than random sampling for identifying long-range redundancies, the quadratic build-time complexity is a primary architectural constraint.
- Greedy Selection vs. Computational Overhead: The decision to utilize non-overlapping motifs was a deliberate trade-off. The marginal compression gain of allowing overlaps

is negated by the pseudo-quadratic cost of re-calculating the global frequency index at each selection step, although this cost can be countered by the usage of Suffix Arrays.

- The "Educated" Reference Hypothesis: The preliminary data suggests that an "educated" reference provides a more stable foundation for random access queries than naive RLZ, as it targets the specific repeating sub-structures (motifs) that define the time series domain.

5.4.1 Future Trajectories in System Optimization

The empirical findings indicate that the randomness of the naive RLZ reference yields a form of "stochastic coverage" that is difficult to fully replicate deterministically within the tested bounds. While custom LF-scored motifs target principled repetitions, random sampling natively captures "accidental" redundancies crucial for maximizing the compression ratio of highly complex time series. Future optimization phases of this architecture should focus on the following vectors:

1. **Addressing Raw Signal Entropy**: A primary driver of motif exhaustion is the inherent noise in floating-point time series, where micro-deviations prevent the exact-match requirements of RLZ. Future pipelines should enforce:

- Delta Encoding: Shifting the analytical focus from raw value drift to repeated increments via a reversible $d[t] = x[t] - x[t-1]$ transform.
- Global Quantization: Introducing a strict quantization grid to collapse microscopic jitter into exact repeats, potentially exposing previously uncaptured multi-scale motifs.

2. **Formalizing Marginal Compression Gain**: Transitioning from the heuristic LF-Score to a formal Marginal Compression Gain objective. This function mathematically prices the bit-cost of RLZ pointers (c_p) against literal symbols (c_ℓ), strictly gating motif inclusion based on net storage savings.
3. **Hybrid Subsequence Selectors**: Engineering a Lazy Greedy selector capable of pulling from both deterministic and stochastic domains. Suffix Array motifs will provide the "stable foundation", while stochastic sampling will backfill the remaining budget to capture accidental redundancies without suffering from exhaustion.
4. **Integration of Diagnostic Pre-checks**: Implementing cheap diagnostic metrics — such as Distinct Substring Slope analysis or Lempel-Ziv complexity proxies — to accurately estimate structural redundancy prior to initiating the computationally expensive reference build sequence.

5.4.2 Credits and Contribution

The architectural benchmarking and C++ integrations detailed in this chapter were conducted during a six-month visiting research period at the University of A Coruña, Spain. This research

was carried out in collaboration with Prof. Antonio Fariña and Dr. Adrián Gómez-Brandón, whose specialized expertise in Compact Data Structures was instrumental to the investigation.

The Python-based software architecture designed to evaluate the STUMPY library and construct the LF-scored references was developed independently as part of this thesis. The high-performance C++ implementation of the RLZ algorithm used to establish the baseline and execute the final compression benchmarks was generously provided by Dr. Adrián Gómez-Brandón, significantly accelerating the empirical validation phase of this study.

CHAPTER 6

Exploratory Directions in Data Integration and LLMs

This chapter transitions from the rigorous empirical benchmarking of temporal models to an exploratory investigation into multi-modal data integration. The projects detailed herein — ranging from judicial prediction to LLM-driven Entity Matching — are presented as positional studies and functional prototypes. Rather than providing exhaustive validation, these sections aim to map the architectural challenges of semantic fusion and to propose methodologies, such as the introduced WYM framework, as paths toward more transparent and explainable data science workflows.

6.1 The Challenge of Judicial Prediction

The research path of this thesis started with exploratory directions in relational and semantic data, catalyzed by an industrial collaboration with the court of Modena. The primary objective of this three-year-long engagement was to develop a predictive framework for criminal trial durations – a task that, at first glance, appeared to be a standard TS regression problem. However, the practical reality of judicial data exposed significant limitations in traditional quantitative approaches and highlighted the necessity for a broader Data Fusion paradigm.

The first year of this project was spent gathering data from the court's I.T. system and analyzing it. The resulting dataset had fields with missing data and fields that needed to be standardized. For example, a few columns could be reconducted to a categorical type, but had arbitrary strings having specific legal acronyms. Some of this data was also not relevant for the task at hand, so it had to be excluded for forecasting; however, even if not directly relevant for the final objective, it was useful for statistically studying the available data to compute some metrics. A few use cases were: (i) analyzing the distribution of defendants to determine the frequency of cases in which multiple people were involved: the number of defendants can directly impact the duration of a trial; (ii) visualizing the distribution of trial durations, and comparing it with the distribution of defendants to study a possible correlation between the two variables; (iii) observing the distribution over the year of case openings, closings and filings, to forecast the staff's workload throughout the months.

The main product of the first year was a 160-pages long report, examining in depth every statistical aspect of the provided dataset.

The most difficult part was standardizing and separating the list of accusations, as they consisted of a string composed of an unstructured list of laws per entry. Each law could present with dots, commas, acronyms, and abbreviations; non-standard formats were arbitrarily chosen, making it even more complicated to create an algorithm that could effectively and efficiently separate each law on a manually predetermined set of separation tokens. At the beginning of this study, tools like LLMs were not yet available as they are today, otherwise this task could have been completed by leveraging the semantic comprehension of one of the many language models available in literature. Moreover, separating each law from the list also meant replicating

all the other fields for that entry, resulting in a higher memory consumption and computational cost for further processing.

Because of a Non Disclosure Agreement with the court, the results of this analysis cannot be shown as they contain sensitive data.

6.1.1 Data Scarcity and Human Unpredictability

The judicial process is inherently characterized by high variance and the presence of a dominant "human factor". The duration of a trial is often dictated not only by the nature of the crime, but by the unpredictable decisions of legal counsel, witnesses, and judges. In this context, the structured data available — primarily consisting of basic case metadata, e.g., the number of defendants, the crimes they were accused of in the form of a list of violated laws, the dates of the case opening, closing and registration — was found to be insufficient for training either DL or ML robust forecasting models. This scarcity created a "predictive ceiling" where models based solely on historical temporal averages failed to capture the nuances of individual proceedings.

6.1.2 The Multi-modal Ensemble Hypothesis

To overcome these limitations, an hypothesis was formulated, focused on Knowledge Extraction from the unstructured data modalities that actually drive the judicial process. A trial's duration could be more accurately modeled through an ensemble of specialized models designed to process disparate data types:

- Acoustic Analysis: Processing audio recordings of witness testimonies to detect sentiment or procedural delays.
- Visual Evidence: Analyzing photos and videos from crime scenes or witness testimonies.
- Semantic Document Parsing: Extracting static features from legal filings and transcripts to engineer multivariate signals.

The proposed architecture would involve a weighting system analogous to the transformer's attention mechanism, where the importance of each modality (e.g., a specific document vs. an audio clip) would be dynamically adjusted based on its relevance to the specific case type. By model each information type as a different kind of embedding, different modules could be specialized to handle that specific kind of data, learning how to transform it or relate it to other embeddings to combine all available information. While the project was eventually constrained by the lack of a sufficient longitudinal dataset for training such complex architectures, the investigation served as a critical precursor to the work done in the field of Entity Matching. It established that before multi-modal signals can be aggregated, a reliable, explainable mechanism must exist to link disparate entities across unstructured and relational domains. Explainability is crucial in many fields, including legal infrustructures, where the motives for specific decisions must be made clear across all the steps of the trial.

6.2 Explainable Entity Matching (EEM) for Data Fusion

The transition from judicial forecasting to broader data integration necessitates a robust mechanism for Semantic Data Fusion – the ability to unify heterogeneous data sources into a single, consistent truth. EM serves as the foundational task in this process, identifying whether disparate data records refer to the same real-world entity. Table 6.1 summarizes the contribution of this thesis to the fields of Entity Matching and Information Retrieval, while also presentig nthe roadmap for the remainder of this section.

Component	Nature of Contribution	Scope
Transformer Analysis	Survey & Positioning	Critical analysis of the "Black-Box" dilemma in contemporary EM models.
WYM Framework	Novel Method	Introduction of Decision Units for intrinsic interpretability.
Recruiting Case Study	Empirical Evaluation	Exploratory application of LLMs for specialized semantic ranking.
RAG Benchmark	Prototype / Positioning	Preliminary investigation into hallucination mitigation in code generation.

Table 6.1: Summary of Contributions for Chapter 6.

6.2.1 From Symbolic to Generative Matching

Historically, EM relied on symbolic engineering and rule-based systems, which offered high transparency but struggled with the inherent ambiguity of natural language. The field has recently undergone a paradigm shift toward transformer-based architectures, which altered the performance baseline of models solving the task, but introduced the "black box" dilemma [Pag+24]. Off-the-shelf transformer models, such as BERT, have set new benchmarks in EM performance by capturing hidden matching patterns through multi-head self-attention mechanisms. However, despite their accuracy, transformers typically operate as opaque classifiers. They limit the user's insight into the motivations behind a match, which is problematic in sensitive domains — such as the legal infrastructures discussed in Section 6.1 — where the justification for a decision is as critical as the decision itself.

6.2.2 The WYM (Why do You Match?) Framework

To bridge the gap between performance and transparency, we introduced **WYM**, an intrinsically explainable model for EM [Bar+23]. Unlike post-hoc explanation methods that approximate a model's behavior, WYM is designed to be interpretable by construction. The framework leverages **Decision Units** (DUs) at its core: the decomposition of entity descriptions into pairs of similar terms across entities or unique terms that differentiate them. These DUs define a new, compact **intrinsically explainable** feature space where matches are justified by specific shared or contrasting terminology. A match between two records e_A and e_B can be formalized as a function of their shared semantic units:

$$\text{Match}(e_A, e_B) = f(\text{DU}(e_A, e_B)) \tag{6.1}$$

Each DU can contribute positively or negatively to a match decision, thus allowing the model to decide whether it's a match or not depending on the match threshold (e.g., 0.5) by summing the contribution score of each DU. For example, a DU composed of two semantically similar words would contribute positively, while an **unpaired** DU, meaning that a token couldn't find its counterpart in the other entity's description, would contribute negatively. This mechanism allows for the generation of customized, human-verifiable explanations that pinpoint exactly which attributes or terms drove the matching decision.

6.2.3 LLMs in the Matching Loop

The evolution of EM continues with the integration of Large Language Models (LLMs), which move beyond binary classification toward generative reasoning.

- LLMs offer the capability for zero-shot semantic linkage, leveraging their vast pre-trained knowledge to understand context without extensive task-specific training.
- Balancing Accuracy and Verifiability: While traditional DL focuses strictly on accuracy, LLMs can be prompted to provide Chain-of-Thought (CoT) explanations. This aligns with the "Explainability" requirement identified in the judicial study, ensuring that data fusion is not only accurate but also auditable by human experts.

This move toward semantic, LLM-driven matching provides the necessary architectural "glue" to link the disparate data modalities encountered in complex real-world tasks, such as the matching of professional profiles in the recruitment domain. An example of application of LLM-driven matching is provided in Section 6.3, where the work carried out with a Master's student, Cyntia Valeria Biondi Russo, exploited Qwen3 to clean data, retrieve relevant information, validate records and create a ranking framework to have a list of Curriculum Vitae (CVs) that could be a good fit for a given Job Posting, ordering the CVs by relevance of the semantic information they report.

6.3 Exploratory Case Study: Semantic Linkage in Recruitment

The potential utility of LLM-driven Entity Matching is explored here through a qualitative case study in the recruitment domain, where data is highly semi-structured and multi-faceted. This section details the exploratory study conducted in collaboration with Cyntia Valeria Biondi Russo, which adapted the principles of semantic data fusion to the recruitment pipeline.

6.3.1 The Recruitment Matching Problem: Beyond Keywords

Traditional recruitment systems often rely on syntactic keyword matching, which fails to capture the semantic nuance of professional experience. A candidate might list "Natural Language Processing" while a job posting requires "LLM Development"; while syntactically different, they are semantically aligned. Job postings and CVs are often written in varying styles, languages, and

formats, requiring a robust "glue" to establish a meaningful link. Therefore, it is a particularly difficult field to work on, especially for systems that cannot capture the semantics of text. The objective was to create a ranking framework that evaluates the semantic relevance of a CV against a specific Job Posting, providing a prioritized list of candidates based on qualitative fit rather than simple word counts.

6.3.2 The Role of Qwen and Generative Reasoning

This prototype utilized Qwen3, a Large Language Model, to investigate the feasibility of end-to-end generative reasoning for parsing and isolating relevant professional information. Qwen was employed to parse unstructured text gathered through a web scraper, removing noise and isolating relevant information such as core skills, seniority levels, and educational background. The model also acted as a semantic validator, ensuring that the extracted entities maintained their contextual meaning throughout the transformation process. Both job postings and CVs were divided by the model in attributes that could be perturbed to generate examples for the model to perform Few-Shot predictions. The model also evaluated the importance of each attribute, thus weighting the semantic distance between job posting and CV attributes. For example, if a job posting clearly stated that candidates were required to have a driving license, the corresponding attribute in the CVs had more importance in comparison to the same attribute for other job postings that didn't explicit this need. The lack of the attribute, or of a driving license, from the candidate's CV lowered its score proportionately. By analyzing the semantic overlap between the job requirements and the candidate's history, the system generated a relevance score used to rank the CVs.

The ultimate task for Qwen was that of providing **explainability**: the scores highlighted and explained the reasons why a CV couldn't be aligned with a job posting, thus giving recruiters the possibility to focus their energy on the most aligned CVs first, then evaluate candidates that could still have somehow aligned skills to the requirements. A similar framework could also propose to automatically reject CVs that are scored below a certain threshold, avoiding the tedious task of filtering each candidate manually.

6.3.3 The Shift to Prompt Engineering

A critical finding of this study was the emergence of Prompt Engineering as a new fundamental task in the data integration workflow. Unlike traditional Language Models (LMs) that require architectural fine-tuning for specific tasks, LLMs are guided by the quality of the input prompt. Effective prompt engineering was required to "teach" the model the criteria for a match, such as how to weight technical skills versus soft skills or how to interpret career gaps. The transition from fixed symbolic logic to prompt-based generative logic represents a shift in how data scientists "program" integration tasks, emphasizing the need for clarity and context in the instructions provided to the LLM.

6.3.4 Application of the "Decision Unit" Concept

The work in this domain serves as a practical validation of the Decision Unit (DU) concept introduced in Chapter 6. In the recruitment context, a DU might be the alignment between a "Python" requirement in the job posting and a "backend developer" title in the CV. By using Qwen to identify and explain these units, the framework provides an explainable ranking, allowing

recruiters to understand why a candidate was deemed a good fit based on specific semantic justifications.

6.4 Prototyping Retrieval Augmented Generation (RAG) Architectures for Information Access

While Entity Matching focuses on the structural integration of data, Retrieval-Augmented Generation (RAG) addresses the challenge of information access and factual accuracy within generative workflows. As discussed in Section 2.4, LLMs often suffer from "hallucinations", generating plausible but factually incorrect information when their internal weights lack specific or up-to-date context. To investigate this, a benchmark software was developed to evaluate the impact of external knowledge retrieval on code generation tasks.

Status Note: The following section describes a proof-of-concept prototype designed to investigate the feasibility of grounding LLMs in external technical documentation. While the results offer promising insights, they are presented as a positional investigation into the architectural necessity of RAG, highlighting the requirements for future production-grade systems rather than providing a definitive empirical validation.

6.4.1 Experimental Infrastructure: PostgreSQL and Vector Search

The core of this investigation involved a comparative study of LLM performance, with a specific focus on Small Language Models (SLMs), when augmented by a RAG pipeline. A vector database was deployed using PostgreSQL (via the pgvector extension) to store high-fidelity documentation (the latest version) for standard data science libraries, including Pandas, NumPy, Matplotlib, and Scikit-learn. By retrieving relevant documentation snippets before generating code, the models were provided with a "frozen" reference of function signatures and parameters, preventing the model from relying on potentially outdated or corrupted internal memory. The infrastructure was based on Marta Santacroce's Master's Thesis framework, which consists of a hybrid methodology designed to address Information Retrieval (IR) tasks through RAG. The code is available at the following link.

Marta's comprehensive RAG system combines document processing, hybrid search, and language model generation for question-answering tasks. The system is built around a PostgreSQL database with pgvector extension for vector storage and retrieval. The system implements three distinct document splitting strategies, each creating chunks with different semantic boundaries while maintaining identical embedding generation and database schema. The recursive splitter uses fallback separators `["\n\n", "\n", " ", ""]`, the character splitter employs a fixed `"\n\n"` separator, and the semantic splitter utilizes a percentile-based breakpoint detection mechanism. All strategies use the **multilingual-e5-large model** for final embeddings, with the semantic splitter additionally employing **multilingual-e5-small** for splitting decisions. Documents are processed page-by-page using **PyMuPDF**, with automatic language detection supporting 22 languages.

The core retrieval mechanism employs a hybrid search approach that combines lexical BM25 full-text search with semantic vector similarity. The system executes parallel searches using PostgreSQL's native full-text search capabilities through the `tsv_content` column and pgvector's cosine similarity operator. The hybrid score is calculated using a weighted combination:

$$\text{final\_rank} = 0.3 \cdot \text{rank\_bm25} + 1.0 \cdot \text{rank\_semantic} \tag{6.2}$$

This approach strongly favors semantic similarity over lexical matching. Results are filtered by a quality threshold of 0.92 and limited to 20 candidates before reranking. A critical refinement step employs a CrossEncoder model (**ms-marco-MiniLM-L-6-v2**) to rerank the top 20 hybrid search results. This reranking process directly scores query-document pairs, enabling cross-attention mechanisms to identify specific relevance patterns that bi-encoders cannot capture. The system returns the top 10 reranked results to the generation component.

Finally, the system features an experimental clustering pipeline that uses the HDBSCAN algorithm to create inter-document clusters, enabling alternative retrieval strategies. This feature extends the database schema with additional columns for cluster IDs and combined embeddings.

6.4.2 Mitigation of Hallucinations and CoT Support

The preliminary results of this small-scale benchmark suggest that even SLMs can achieve improved code accuracy within the specific scope of this PostgreSQL-backed documentation retrieval task when grounded by a factual reference. The prototype suggests that a RAG pipeline can reduce the frequency of "hallucinated" or deprecated API calls by providing a factual "anchor" for the model's logical derivation.

While Chain-of-Thought (CoT) reasoning was not explicitly tested in this iteration, existing literature suggests RAG provides substantial aid to such logical processes.

6.4.3 Conclusions, Contributions and Future Work

Despite these exploratory gains, this study highlights a critical need for software infrastructure robustness. The primary methodological finding of this prototype is that the reliability of generative results is intrinsically tied to the "sanity" of the retrieval mechanism itself.Consequently, the current framework serves to identify the sanity-check requirements necessary for future production-grade systems.

Future research must focus on verifying that infrastructure components — ranging from embedding generation to similarity thresholds — are statistically sound to ensure that performance reflects architectural gains rather than artifacts of a specific data slice.

The most impactful research in this section was the report drawn up for the court of Modena, which provided meaningful insights into the court's workload. It allowed the staff to statistically visualize the internal processes of the court, providing possible explanations for occasional bottlenecks, the most critical being the one caused by the COVID-19 pandemic in 2020.

CHAPTER 7

Synthesis and Final Conclusions

This thesis has explored the challenge of information discovery through three interconnected lenses:

1. Temporal Discovery: Establishing a rigorous, "SOTA-chase" free framework to benchmark the recovery of signals from irregularly sampled and sparse datasets.
2. Efficient Infrastructure: Investigating deterministic motif discovery to optimize queryable compression for massive temporal data lakes.
3. Semantic Integration: Evaluating the role of Large Language Models (LLMs) and EEM in bridging the gap between structured records and unstructured context.

7.1 Unified Findings: Trustworthiness and Interdisciplinary Innovation

The central finding of this work is that specialized domains require more than just accuracy; they require verifiable reliability. For example, in Time Series, it was discovered that low error metrics (MAE/MSE) often masks a failure to learn underlying dynamics, a gap revealed by negative R^2 scores in sparse scenarios. This mirrors the "black-box" dilemma in Entity Matching, where a correct match is insufficient without a human-verifiable justification.

A significant theme of this research has been the application of disparate techniques to solve niche problems. For instance, applying STUMPY — a tool traditionally used for Time Series motif discovery — to optimize Relative Lempel-Ziv (RLZ) reference construction demonstrates how unconventional viewpoints can unlock new infrastructure strategies.

7.2 Future Work: Toward an Intelligent Data Ecosystem

7.2.1 Meta-Learning for Automated Imputation

As proposed in the ISTS benchmark evolution, a future "intelligent" layer could automate the choice between simple (e.g., LOCF) and complex (e.g., BRITS) imputation methods. Essentially, it would consist in "learning how models learn", to tailor the pipeline to the forecasting model depending on the type of input. This could be operationalized via:

- Simple Classifiers: A lightweight model trained to predict the optimal imputation strategy based on the current dataset's sparsity and the intended downstream forecasting model.

- Random Forest Architectures: Utilizing specific patterns and data formats extracted from the signal to decide the most efficient processing path, balancing computational cost against predictive gain.

7.2.2 Compression-Aware Deep Learning

A compelling exploratory direction is the potential for models to learn directly from compressed data structures. Given that some Compact Data Structures rely on matrixes and matrix operations, there is a theoretical opportunity to let DL models "unravel" information from compressed representations. Storing information in a drastically smaller structure and training directly on it would revolutionize the "Efficiency Pillar", merging storage and learning into a single high-performance layer.

7.2.3 Multi-modal Foundations

Building on the judicial prediction hypothesis, future work should continue exploring the ensemble-based attention mechanism. By integrating acoustic, visual, and semantic embeddings into a unified weighting system, it could be possible to move closer to autonomous data engineering that respects the complexities of human-centric domains like law and recruitment.

7.2.4 Proposal for a more efficient future

While the "SOTA-chase" often prioritizes marginal accuracy gains, this thesis argues for a return to architectural integrity. Whether through identifying the "Top 3" imputation methods that provide a faithful signal or engineering "educated" references for compression, the ultimate goal is an integrated data infrastructure that is efficient, transparent, and robust against real-world imperfections.

Technology should be standardized to enable algorithms from different fields to interoperate. An example of this is a system where Suffix Array motifs serve as contextual anchors for RAG-powered systems, and where the "Meta-Learning" layer automatically optimizes the path from raw storage to explainable insight. This approach is already being put to practice in a way in ensemble models like Mixtral [Jia+24], where a Gating Network chooses which expert to use at each layer of the model so to always perform at its best depending on the input information itself. By bridging the gap between numeric compression and semantic reasoning, it is possible to provide a foundation for discovery that is both computationally lean and human-verifiable.

7.3 Credits

English version. European Union-funded doctoral thesis - Next Generation EU, Mission 4, Component 1 "Strengthening the provision of educational services: from nursery schools to universities" - Investment 4.1 "Expanding the number of research doctorates and innovative doctorates for public administration and cultural heritage".

Italian version. Tesi di dottorato finanziata dall'Unione europea - Next Generation EU, Missione 4, componente 1 "Potenziamento dell'offerta dei servizi di istruzione: dagli asili nido all'Università" - Investimento 4.1 "Estensione del numero di dottorati di ricerca e dottorati innovativi per la pubblica amministrazione e il patrimonio culturale".

Appendix A: A Systematic Framework for the Evaluation of Irregularly Sampled Time Series Forecasting Workflows

This appendix provides the comprehensive results of the experimental campaign conducted across 2800 individual runs for RQ1.1 and 600 for RQ2.1. The data reported here supports the findings discussed in Chapter 3, offering a granular view of model performance across varying levels of data missingness, temporal horizons, and architectural configurations.

A.1 Standardization and Metrics

To ensure comparability across heterogeneous domains — ranging from high-frequency energy data (ETTh1) to stochastic financial streams (ExchangeRate) and healthcare indicators (ILI) — all metrics are reported in the standardized space: (i) Mean Absolute Error (MAE) / Mean Squared Error (MSE), error metrics where lower values indicate higher reconstruction or forecasting fidelity, and (ii) R^2 Score (Coefficient of Determination), used to identify "failure regimes", where negative values indicate the model failed to capture the underlying data dynamics.

A.2 Visualization Reference

Every data point in the following sections represents the mean of five independent runs (Seeds 0-4).

- Critical Difference (CD) Diagrams: These diagrams visualize the results of the Friedman test and the Nemenyi post-hoc analysis. Horizontal bars (cliques) connect models whose performance differences are not statistically significant at a confidence level of $\alpha = 0.05$.
- Radar Plots: Illustrate cross-domain sensitivity; models closer to the origin (center) represent superior performance for error metrics.
- R^2 Quadrant Plots: Scatter plots used to categorize model reliability, specifically focusing on the transition from failure (Quadrant IV) to the target zone (Quadrant II).
- Grouped Bar Plots: Compare families of methods (e.g., Simple vs. Complex imputation) across the three primary sparsity levels: 20%, 50%, and 80% NaN.
- RPI Plots: Relative Performance Index plots, these bar plots illustrate the percentage-based gain (or loss) in error reduction when utilizing Native ISTS models compared to Imputation + Standard pipelines. Positive percentages indicate the superiority of the Native approach at specific sparsity levels (20%, 50%, 80%).

A.3 RQ1.1

A.3.1 Raw Performance Metrics

Table A.1, Table A.2 and Table A.3 report the mean values for MAE, MSE, and R^2 across all datasets, horizons, and sparsity levels (20%, 50%, 80%) averaged over 5 runs.

Forecasting Dataset		NaN Percentage	Model	20%								50%								80%							
				Deep-AR	Dlin-ear	GRU-D	mTAN	N-HiTS	Patch-TST	Prime-Net	TCN	Deep-AR	Dlin-ear	GRU-D	mTAN	N-HiTS	Patch-TST	Prime-Net	TCN	Deep-AR	Dlin-ear	GRU-D	mTAN	N-HiTS	Patch-TST	Prime-Net	TCN
Beijing-PM2.5		96		0.768	0.771	0.852	0.765	0.781	0.763	0.771	0.807	0.768	0.771	0.845	0.751	0.782	0.768	0.743	0.767	0.762	0.769	0.826	0.761	0.769	0.783	0.738	0.764
		192		0.768	0.770	0.864	0.761	0.781	0.793	0.765	0.796	0.768	0.770	0.856	0.761	0.785	0.795	0.765	0.779	0.763	0.768	0.829	0.751	0.786	0.801	0.745	0.760
		336		0.767	0.765	0.859	0.753	0.792	0.751	0.788	0.840	0.767	0.764	0.873	0.760	0.791	0.760	0.763	0.807	0.762	0.763	0.862	0.750	0.774	0.759	0.747	0.774
		720		0.767	0.763	0.872	0.752	0.793	0.763	0.761	0.773	0.767	0.764	0.876	0.757	0.780	0.776	0.748	0.762	0.762	0.762	0.849	0.756	0.769	0.784	0.758	0.761
ETTh1		96		0.955	0.312	0.474	0.350	0.300	0.301	0.365	0.396	0.946	0.337	0.471	0.352	0.299	0.302	0.366	0.368	0.942	0.607	0.434	0.380	0.300	0.307	0.378	0.374
		192		0.949	0.281	0.420	0.359	0.290	0.289	0.352	0.383	0.940	0.287	0.443	0.353	0.303	0.290	0.336	0.321	0.937	0.363	0.403	0.369	0.303	0.293	0.353	0.355
		336		0.938	0.299	0.374	0.343	0.300	0.291	0.342	0.324	0.931	0.346	0.341	0.343	0.298	0.288	0.336	0.341	0.928	0.710	0.391	0.336	0.317	0.299	0.346	0.330
		720		0.875	0.319	0.620	0.395	0.330	0.368	0.381	0.387	0.871	0.329	0.532	0.368	0.308	0.368	0.367	0.380	0.871	0.418	0.419	0.355	0.358	0.370	0.398	0.455
Exchange-Rate		96		1.593	0.515	0.834	0.729	0.618	0.373	1.007	0.723	1.607	0.770	0.845	0.778	0.534	0.378	0.888	0.718	1.583	1.244	0.950	0.858	0.359	0.382	0.860	0.762
		192		1.592	0.634	1.277	0.842	1.323	0.497	0.951	0.868	1.605	0.677	1.372	0.918	0.861	0.506	0.994	0.868	1.582	0.870	1.334	0.966	0.726	0.495	1.007	0.801
		336		1.612	1.071	1.088	0.758	1.036	0.642	0.817	0.772	1.601	1.363	0.997	0.850	0.852	0.644	0.811	0.811	1.578	2,016	0.962	0.999	0.883	0.639	0.943	0.722
		720		1.545	1.351	1.751	0.889	0.868	0.926	1.585	1.644	1.558	1.489	1.378	0.914	0.876	0.926	1.402	1.240	1.537	1.402	1.251	1.036	0.987	0.923	1.321	1.338
French-Piezo		96		0.284	0.019	0.128	0.041	0.017	0.017	0.021	0.020	0.284	0.018	0.136	0.039	0.017	0.017	0.021	0.018	0.285	0.020	0.102	0.029	0.026	0.017	0.024	0.020
		192		0.282	0.022	0.070	0.040	0.024	0.020	0.026	0.028	0.283	0.022	0.253	0.035	0.027	0.020	0.025	0.026	0.284	0.026	0.189	0.030	0.022	0.021	0.026	0.029
		336		0.284	0.017	0.157	0.037	0.021	0.014	0.022	0.021	0.285	0.016	0.151	0.030	0.022	0.015	0.022	0.021	0.285	0.018	0.090	0.031	0.040	0.015	0.021	0.075
		720		0.288	0.020	0.085	0.042	0.016	0.016	0.037	0.187	0.288	0.021	0.156	0.043	0.019	0.016	0.026	0.244	0.288	0.021	0.062	0.042	0.022	0.017	0.034	0.151
ILI		24		2.672	2.671	2.005	2.002	1.053	0.674	2.033	1.295	2.752	2.768	2.111	2.084	1.052	0.744	2.217	1.386	2.800	2.813	2.224	2.463	1.055	1.001	2.458	1.118
		36		2.658	2.646	1.798	2.038	1.027	0.696	2.059	1.219	2.749	2.758	1.858	2.115	0.944	0.792	2.281	1.282	2.799	2.934	1.821	2.456	1.091	1.034	2.496	1.152
		48		2.631	2.451	1.596	1.890	1.210	0.759	2.015	1.185	2.734	2.542	1.666	2.076	1.147	0.846	2.191	1.241	2.797	2.572	1.517	2.416	1.088	1.076	2.414	1.401
		60		2.598	2.469	1.954	1.792	1.255	0.938	1.869	1.577	2.734	2.579	1.931	1.923	1.306	1.078	1.979	1.594	2.795	2.617	1.816	2.396	1.331	1.360	2.319	1.586
USHCN		24		0.766	0.384	0.402	0.580	0.398	0.438	0.455	0.382	0.766	0.389	0.407	0.492	0.394	0.449	0.442	0.385	0.765	0.409	0.393	0.464	0.394	0.487	0.597	0.383
		36		0.766	0.429	0.425	0.511	0.434	0.528	0.506	0.419	0.765	0.434	0.427	0.603	0.427	0.537	0.535	0.412	0.765	0.458	0.420	0.592	0.421	0.592	0.585	0.414
		48		0.766	0.466	0.453	0.654	0.453	0.616	0.600	0.432	0.766	0.474	0.443	0.644	0.449	0.629	0.651	0.435	0.765	0.503	0.451	0.587	0.456	0.703	0.652	0.442
		60		0.767	0.519	0.522	0.642	0.509	0.730	0.654	0.471	0.766	0.526	0.484	0.611	0.494	0.745	0.690	0.468	0.766	0.551	0.477	0.689	0.498	0.816	0.695	0.473

Table A.1: Raw MAE values gathered for RQ1.1 averaged across 5 runs.

NaN Percentage				Model																					
				20%								50%							80%						
				Deep-AR	Dlin-ear	GRU-D	mTAN	N-HiTS	Patch-TST	Prime-Net	TCN	Deep-AR	Dlin-ear	GRU-D	mTAN	N-HiTS	Patch-TST	Prime-Net	TCN	Deep-AR	Dlin-ear	GRU-D	mTAN	N-HiTS	Patch-TST	Prime-Net	TCN
Dataset	Forecasting Horizon																								
Beijing-PM2.5	96			1.152	1.160	1.524	1.141	1.211	1.308	1.215	1.393	1.154	1.163	1.532	1.122	1.200	1.322	1.094	1.281	1.149	1.162	1.466	1.132	1.162	1.376	1.070	1.200
	192			1.152	1.141	1.581	1.124	1.195	1.348	1.187	1.328	1.154	1.145	1.583	1.124	1.215	1.354	1.187	1.257	1.149	1.145	1.454	1.108	1.187	1.380	1.103	1.164
	336			1.152	1.135	1.556	1.118	1.225	1.227	1.245	1.504	1.154	1.134	1.615	1.130	1.209	1.250	1.163	1.375	1.149	1.139	1.550	1.125	1.172	1.267	1.115	1.212
	720			1.152	1.142	1.584	1.128	1.255	1.335	1.197	1.233	1.154	1.145	1.627	1.141	1.168	1.360	1.145	1.196	1.149	1.146	1.552	1.143	1.162	1.367	1.165	1.195
ETTh1	96			1.081	0.157	0.366	0.183	0.143	0.149	0.196	0.247	1.066	0.178	0.354	0.183	0.143	0.149	0.198	0.207	1.062	0.565	0.283	0.234	0.148	0.153	0.219	0.216
	192			1.069	0.131	0.302	0.190	0.134	0.135	0.183	0.235	1.053	0.135	0.311	0.189	0.144	0.136	0.173	0.157	1.051	0.205	0.250	0.212	0.145	0.137	0.195	0.190
	336			1.047	0.138	0.212	0.174	0.137	0.133	0.172	0.161	1.035	0.188	0.181	0.181	0.139	0.130	0.174	0.185	1.035	0.823	0.232	0.168	0.155	0.142	0.182	0.164
	720			0.930	0.152	0.545	0.227	0.175	0.224	0.218	0.238	0.924	0.161	0.434	0.200	0.153	0.223	0.207	0.219	0.930	0.259	0.267	0.192	0.211	0.224	0.250	0.332
Exchange-Rate	96			3.105	0.403	0.950	0.763	0.572	0.206	1.523	0.781	3.151	1.023	0.977	0.873	0.446	0.211	1.088	0.789	3.072	2.736	1.381	1.031	0.204	0.222	1.042	0.872
	192			3.101	0.530	2.405	1.015	2.536	0.376	1.272	1.017	3.148	0.612	2.720	1.194	1.155	0.385	1.346	1.028	3.067	1.092	2.706	1.321	0.856	0.372	1.399	0.917
	336			3.166	1.562	1.710	0.789	1.565	0.692	0.924	0.838	3.133	2.540	1.487	1.033	1.097	0.699	0.962	0.994	3.054	5.290	1.316	1.396	1.248	0.680	1.240	0.741
	720			2.950	2.413	4.011	1.150	0.990	1.161	3.043	3.872	2.995	2.804	2.456	1.260	1.020	1.163	2.557	2.087	2.922	2.588	2.140	1.643	1.399	1.168	2.412	2.331
French-Piezo	96			0.914	0.016	0.082	0.024	0.017	0.015	0.015	0.020	0.909	0.013	0.104	0.067	0.014	0.012	0.015	0.014	0.917	0.015	0.083	0.019	0.020	0.014	0.018	0.017
	192			0.914	0.027	0.047	0.035	0.024	0.027	0.029	0.031	0.909	0.027	0.241	0.033	0.022	0.024	0.027	0.029	0.917	0.033	0.136	0.030	0.023	0.030	0.026	0.033
	336			0.914	0.027	0.116	0.038	0.063	0.030	0.027	0.029	0.909	0.032	0.093	0.035	0.382	0.039	0.029	0.030	0.917	0.040	0.065	0.036	0.483	0.040	0.034	0.205
	720			0.914	0.019	0.069	0.040	0.016	0.017	0.036	0.556	0.909	0.023	0.126	0.045	0.023	0.019	0.039	0.732	0.917	0.029	0.057	0.052	0.028	0.025	0.047	0.393
ILI	24			8.551	10.025	5.194	5.270	1.498	0.648	5.351	2.229	9.079	10.583	5.751	5.688	1.593	0.759	6.289	2.511	9.480	10.431	6.649	7.723	1.669	1.356	7.703	1.712
	36			8.474	8.970	4.370	5.405	1.361	0.662	5.508	1.936	9.060	9.827	4.710	5.833	1.224	0.820	6.646	2.146	9.475	11.179	4.837	7.701	1.621	1.469	7.912	1.925
	48			8.334	7.191	3.324	4.707	1.805	0.767	5.327	1.817	8.978	7.738	3.664	5.595	1.722	0.932	6.086	1.928	9.460	8.071	3.169	7.494	1.571	1.471	7.470	2.782
	60			8.158	7.452	5.131	4.282	1.954	1.147	4.614	3.272	8.978	8.130	4.930	4.873	2.135	1.497	5.058	3.372	9.452	8.569	4.607	7.382	2.224	2.489	6.909	3.421
USHCN	24			0.876	0.240	0.266	0.533	0.261	0.324	0.328	0.240	0.875	0.246	0.275	0.403	0.258	0.341	0.312	0.245	0.875	0.273	0.257	0.342	0.256	0.398	0.570	0.241
	36			0.876	0.295	0.300	0.418	0.310	0.455	0.415	0.288	0.875	0.302	0.302	0.563	0.300	0.471	0.452	0.277	0.875	0.337	0.292	0.566	0.296	0.567	0.521	0.279
	48			0.876	0.348	0.338	0.693	0.340	0.598	0.548	0.305	0.875	0.359	0.323	0.612	0.335	0.625	0.641	0.311	0.875	0.403	0.337	0.552	0.345	0.761	0.642	0.320
	60			0.877	0.423	0.444	0.656	0.422	0.804	0.648	0.360	0.876	0.433	0.381	0.607	0.397	0.836	0.715	0.355	0.875	0.476	0.377	0.800	0.406	0.984	0.725	0.363

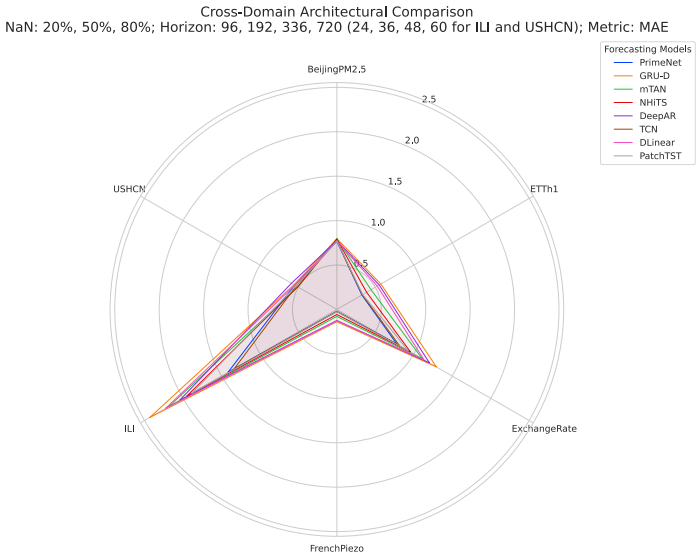
Table A.2: Raw MSE values gathered for RQ1.1 averaged across 5 runs.

Dataset		NaN Percentage	Model	20%							50%							80%						
				Deep-AR	Dlin-ear	GRU-D	mTAN	N-HiTS	Patch-TST	Prime-Net	TCN	Deep-AR	Dlin-ear	GRU-D	mTAN	N-HiTS	Patch-TST	Prime-Net	TCN	Deep-AR	Dlin-ear	GRU-D	mTAN	N-HiTS	Patch-TST	Prime-Net	TCN
ForecastingDataset	Beijing-PM2.5	96		0.000	−0.007	−0.323	0.010	−0.052	−0.136	−0.055	−0.210	0.000	−0.008	−0.328	0.028	−0.040	−0.146	0.052	−0.110	0.000	−0.011	−0.276	0.015	−0.012	−0.197	0.069	−0.044
		192		0.000	0.009	−0.373	0.024	−0.037	−0.171	−0.030	−0.153	0.000	0.007	−0.372	0.026	−0.053	−0.174	−0.028	−0.089	0.000	0.003	−0.266	0.036	−0.033	−0.201	0.041	−0.013
		336		0.000	0.015	−0.351	0.029	−0.064	−0.065	−0.081	−0.306	0.000	0.018	−0.399	0.021	−0.048	−0.084	−0.008	−0.191	0.000	0.009	−0.349	0.021	−0.020	−0.102	0.029	−0.055
		720		0.000	0.008	−0.375	0.021	−0.090	−0.160	−0.040	−0.071	0.000	0.007	−0.410	0.012	−0.012	−0.179	0.008	−0.036	0.000	0.002	−0.350	0.006	−0.012	−0.190	−0.014	−0.040
	ETTh1	96		−5.318	0.083	−1.135	−0.066	0.167	0.128	−0.146	−0.444	−5.170	−0.031	−1.045	−0.060	0.170	0.135	−0.148	−0.197	−5.029	−2.225	−0.603	−0.327	0.162	0.134	−0.241	−0.221
		192		−5.251	0.234	−0.764	−0.110	0.219	0.209	−0.067	−0.374	−5.096	0.218	−0.795	−0.094	0.167	0.214	−0.004	0.092	−4.963	−0.169	−0.414	−0.195	0.176	0.223	−0.103	−0.081
		336		−5.119	0.195	−0.242	−0.014	0.200	0.226	−0.004	0.061	−4.992	−0.083	−0.049	−0.047	0.197	0.248	−0.006	−0.069	−4.871	−3.694	−0.320	0.048	0.118	0.193	−0.031	0.068
		720		−4.437	0.114	−2.176	−0.325	−0.020	−0.307	−0.276	−0.385	−4.349	0.070	−1.509	−0.154	0.118	−0.291	−0.202	−0.269	−4.277	−0.462	−0.507	−0.086	−0.188	−0.273	−0.408	−0.894
	Exchange-Rate	96		−4.564	0.278	−0.703	−0.366	−0.027	0.631	−1.725	−0.401	−4.599	−0.810	−0.732	−0.547	0.213	0.625	−0.927	−0.405	−4.522	−3.898	−1.487	−0.851	0.634	0.601	−0.876	−0.559
		192		−4.557	0.050	−3.316	−0.818	−3.537	0.327	−1.277	−0.817	−4.593	−0.088	−3.839	−1.117	−1.059	0.317	−1.389	−0.821	−4.514	−0.976	−3.858	−1.369	−0.527	0.331	−1.510	−0.629
		336		−4.674	−1.802	−2.061	−0.415	−1.798	−0.240	−0.657	−0.497	−4.566	−3.543	−1.645	−0.834	−0.945	−0.240	−0.703	−0.762	−4.489	−8.582	−1.365	−1.508	−1.223	−0.223	−1.224	−0.342
		720		−4.286	−3.332	−6.214	−1.061	−0.774	−1.080	−4.451	−5.914	−4.319	−3.991	−3.395	−1.226	−0.818	−1.064	−3.532	−2.713	−4.252	−3.704	−2.888	−1.947	−1.496	−1.101	−3.335	−3.194
	French-Piezo	96		0.000	0.983	0.910	0.974	0.981	0.983	0.983	0.978	0.000	0.986	0.884	0.926	0.985	0.986	0.984	0.984	0.000	0.984	0.911	0.979	0.978	0.985	0.980	0.982
		192		0.000	0.970	0.948	0.962	0.973	0.971	0.968	0.966	0.000	0.970	0.735	0.964	0.976	0.973	0.970	0.968	0.000	0.964	0.851	0.968	0.974	0.967	0.972	0.964
		336		0.000	0.970	0.873	0.959	0.931	0.967	0.971	0.968	0.000	0.965	0.898	0.961	0.582	0.957	0.968	0.967	0.000	0.956	0.929	0.961	0.467	0.956	0.963	0.770
		720		0.000	0.979	0.925	0.956	0.982	0.981	0.961	0.391	0.000	0.975	0.862	0.951	0.975	0.979	0.957	0.194	0.000	0.969	0.938	0.943	0.969	0.973	0.948	0.580
	ILI	24		−5.086	−6.207	−2.698	−2.751	−0.065	0.539	−2.809	−0.580	−5.087	−6.418	−2.856	−2.800	−0.059	0.482	−3.205	−0.679	−5.206	−6.909	−3.407	−4.002	−0.021	0.149	−3.980	−0.141
		36		−5.032	−5.399	−2.105	−2.844	0.030	0.529	−2.929	−0.368	−5.075	−5.498	−2.152	−2.906	0.184	0.454	−3.438	−0.440	−5.202	−6.027	−2.051	−3.976	−0.023	0.094	−4.098	−0.283
		48		−4.933	−4.110	−1.368	−2.366	−0.282	0.453	−2.789	−0.284	−5.022	−4.248	−1.455	−2.745	−0.155	0.378	−3.067	−0.293	−5.193	−4.451	−1.074	−3.847	−0.004	0.052	−3.837	−0.728
		60		−4.808	−4.265	−2.649	−2.048	−0.392	0.182	−2.284	−1.323	−5.023	−4.473	−2.310	−2.275	−0.429	−0.002	−2.385	−1.250	−5.187	−4.647	−1.974	−3.778	−0.454	−0.547	−3.508	−1.199
	USHCN	24		−0.002	0.725	0.695	0.389	0.701	0.629	0.625	0.725	−0.003	0.718	0.685	0.538	0.704	0.610	0.643	0.719	−0.003	0.687	0.706	0.608	0.707	0.543	0.348	0.724
		36		−0.003	0.662	0.656	0.522	0.645	0.479	0.525	0.670	−0.003	0.654	0.653	0.355	0.656	0.460	0.482	0.683	−0.002	0.613	0.665	0.351	0.661	0.350	0.403	0.680
		48		−0.003	0.602	0.613	0.206	0.611	0.315	0.372	0.650	−0.003	0.589	0.630	0.298	0.616	0.284	0.266	0.644	−0.003	0.539	0.614	0.368	0.604	0.128	0.263	0.633
		60		−0.004	0.515	0.491	0.249	0.517	0.079	0.258	0.588	−0.003	0.504	0.563	0.305	0.545	0.042	0.180	0.593	−0.003	0.455	0.568	0.082	0.535	−0.128	0.169	0.584

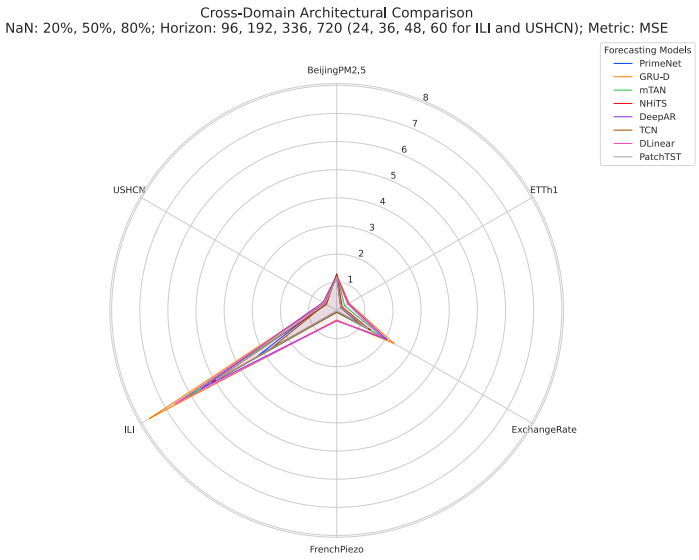
Table A.3: Raw R^2 Score values gathered for RQ1.1 averaged across 5 runs.

A.3.2 Cross-Domain Radar Analysis

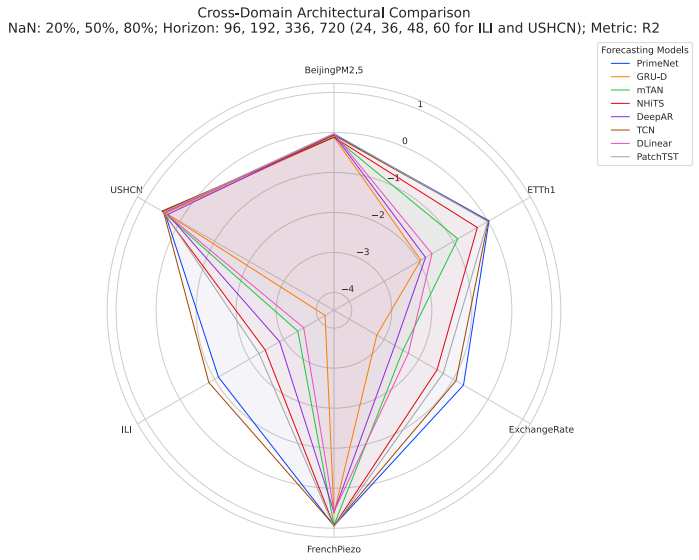
Radar plots illustrate model performance across the six primary domains (BeijingPM2.5, ETTh1, ExchangeRate, FrenchPiezo, ILI, USHCN). First of all, plots where the metrics were averaged over missing data percentages are shown; subsequently, each plot represents a specific NaN% and metric.



(a)

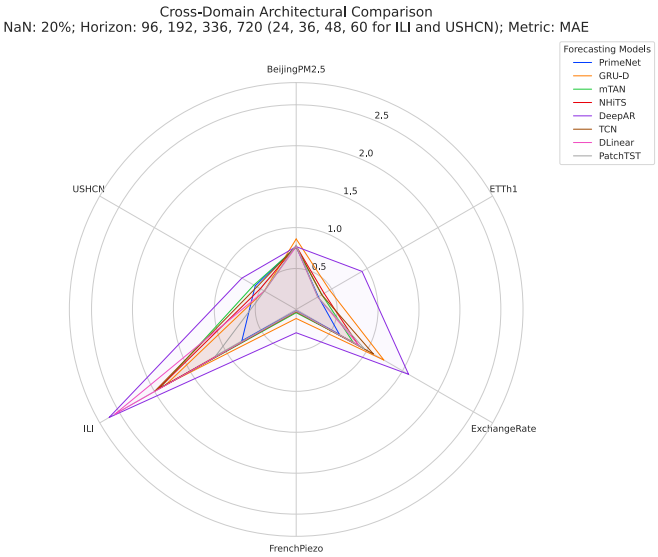


(b)

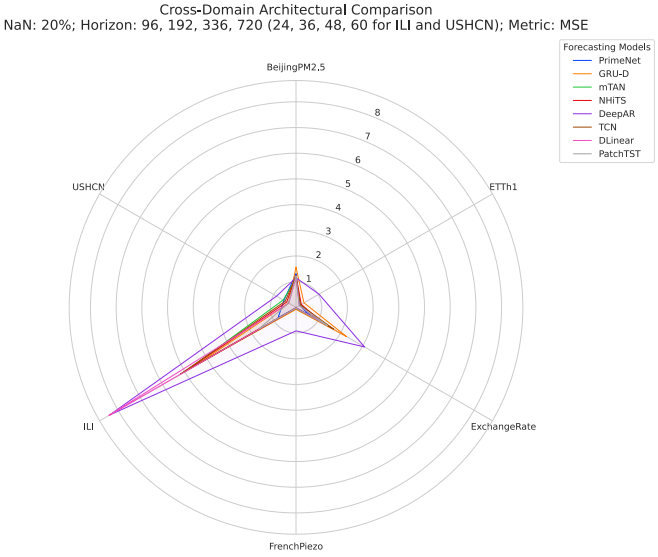


(c)

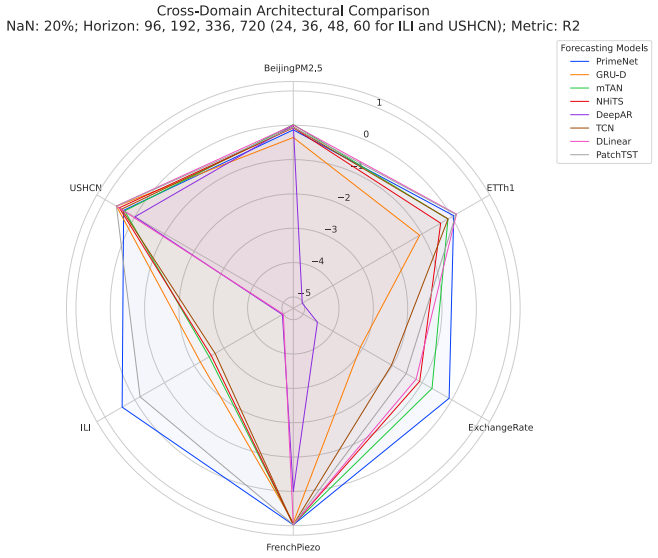
Figure A.1: Radar plot over MAE, MSE and R^2 Score metrics averaging over all missing data percentages and forecasting horizons.



(a)

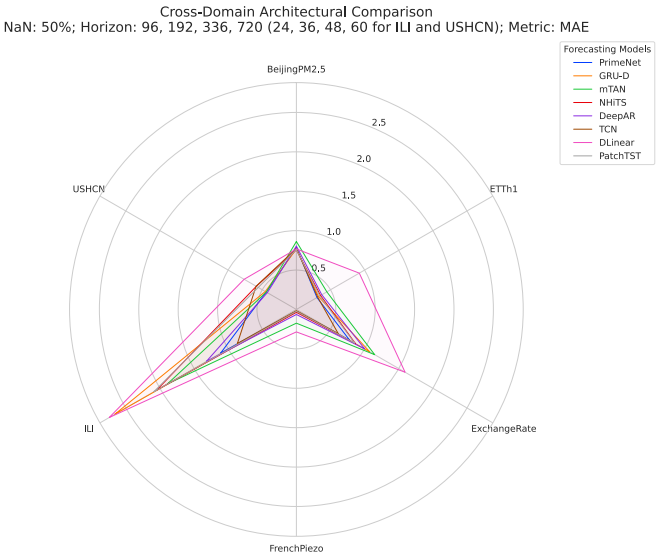


(b)

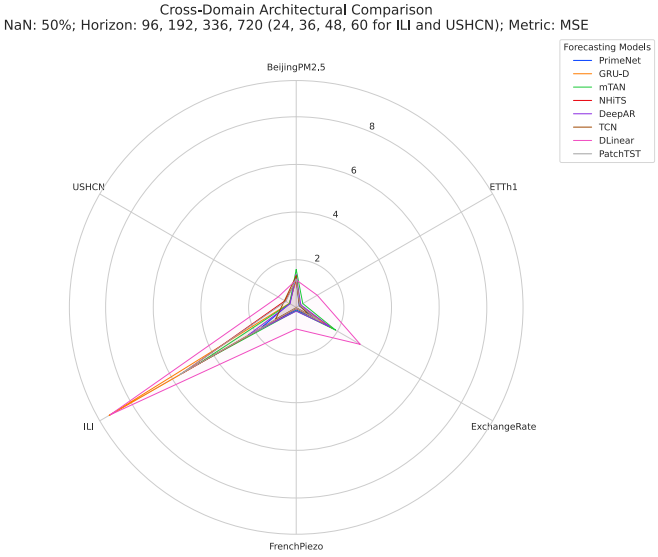


(c)

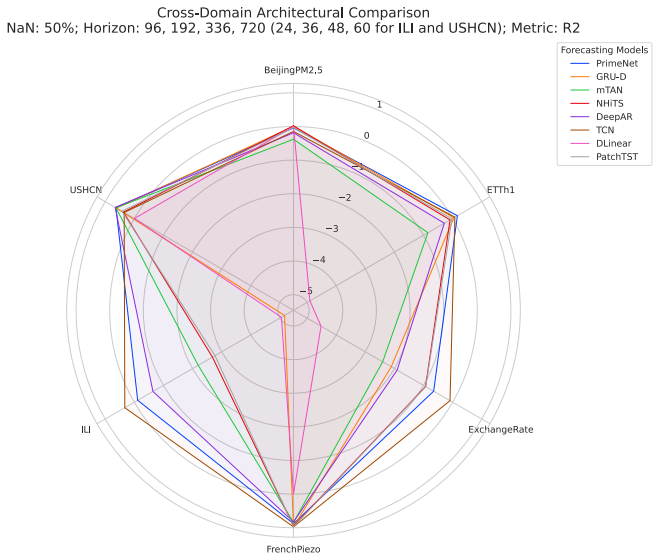
Figure A.2: Radar plot over MAE, MSE and R^2 Score metrics averaging over 20% missing data and all forecasting horizons.



(a)

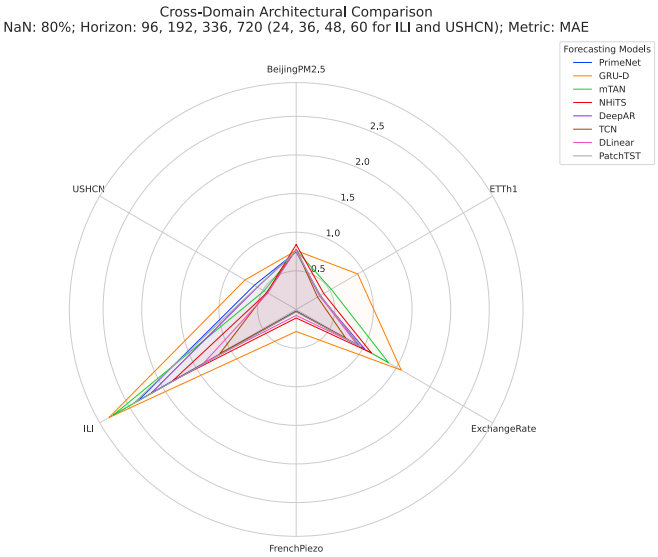


(b)

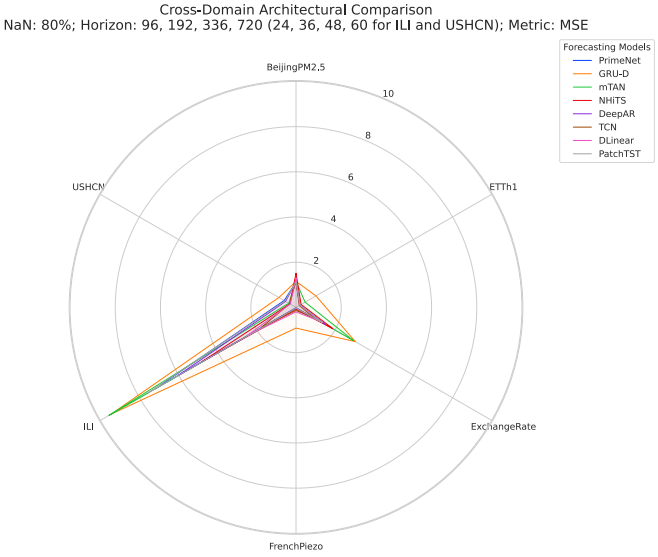


(c)

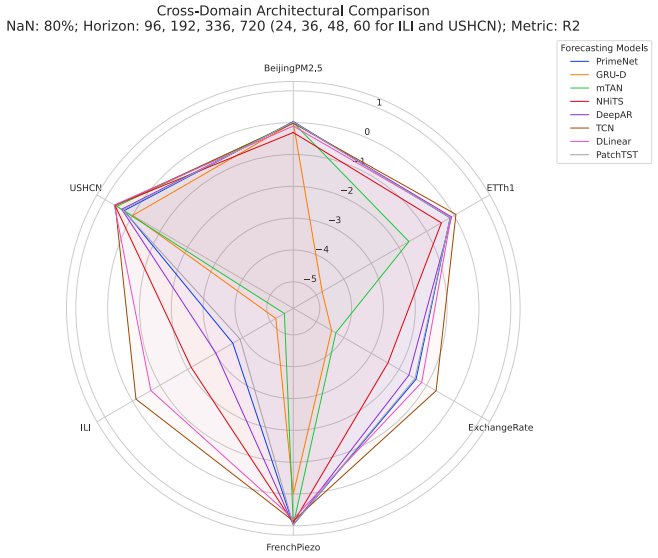
Figure A.3: Radar plot over MAE, MSE and R^2 Score metrics averaging over 50% missing data and all forecasting horizons.



(a)

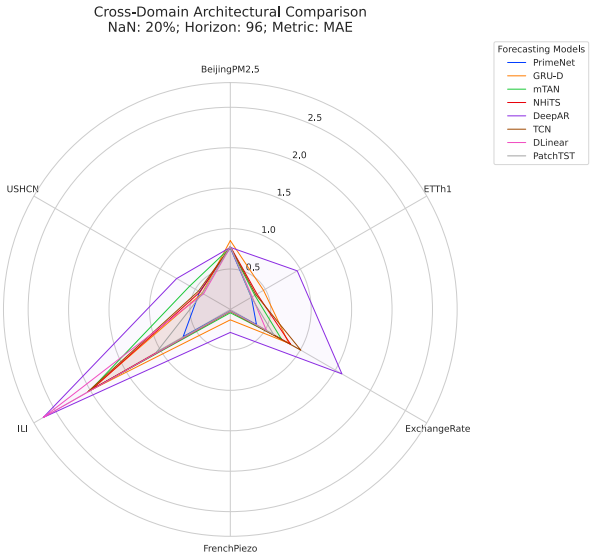


(b)

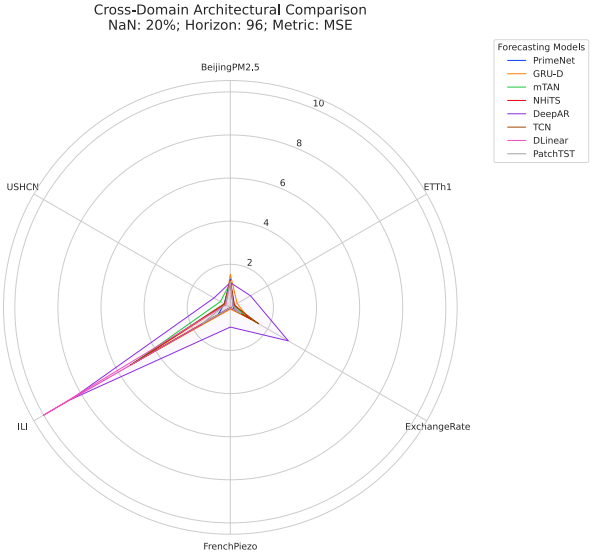


(c)

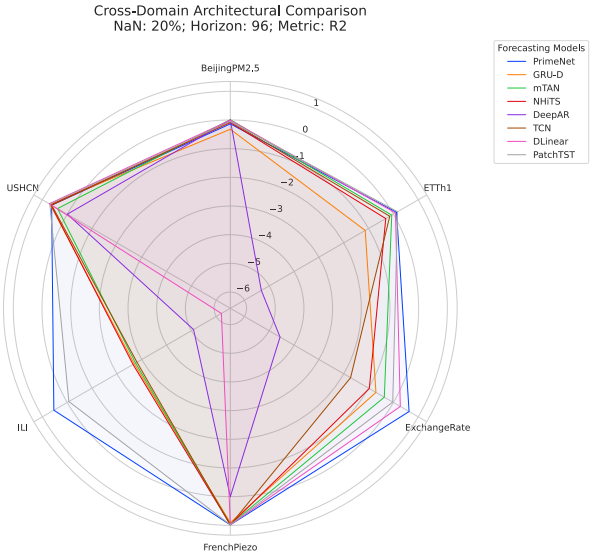
Figure A.4: Radar plot over MAE, MSE and R^2 Score metrics averaging over 80% missing data and all forecasting horizons.



(a)

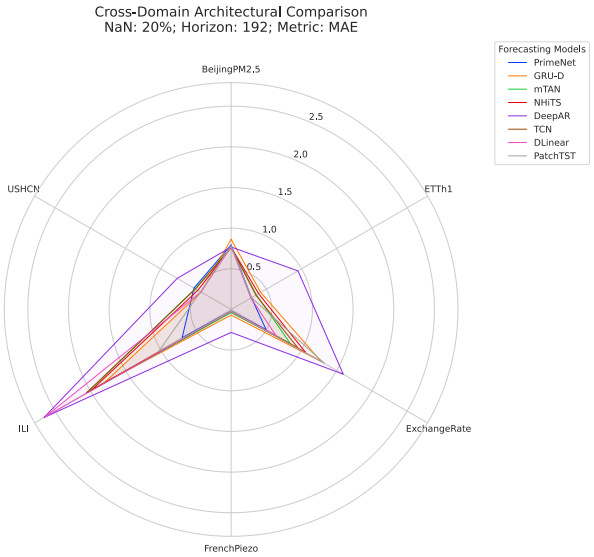


(b)

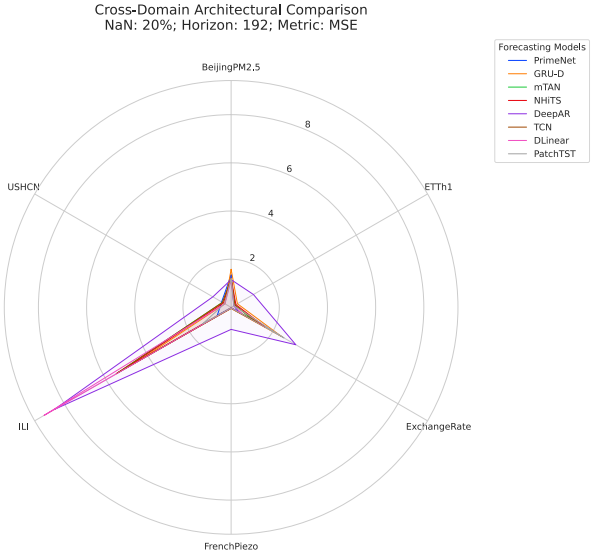


(c)

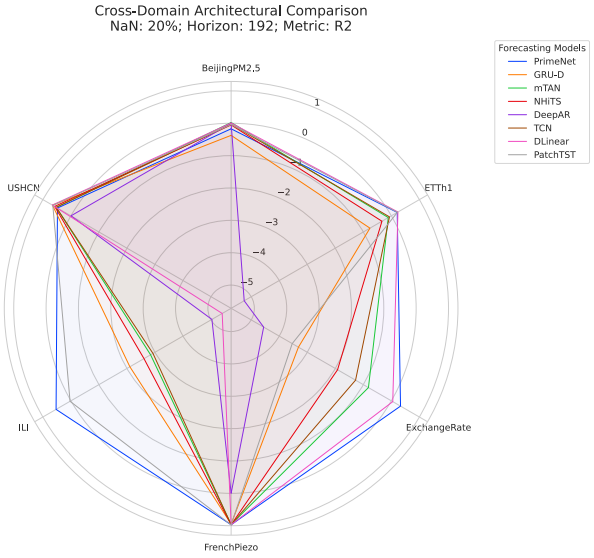
Figure A.5: Radar plot over MAE, MSE and R^2 Score metrics averaging over 20% missing data and a forecasting horizon of 96 (24 for ILI and USHCN).



(a)

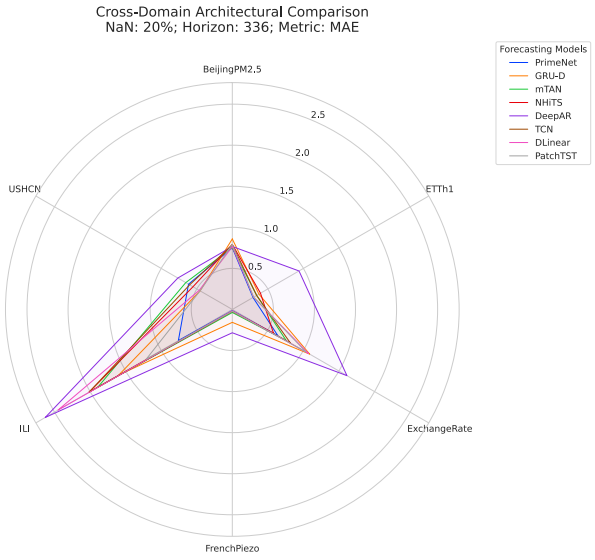


(b)

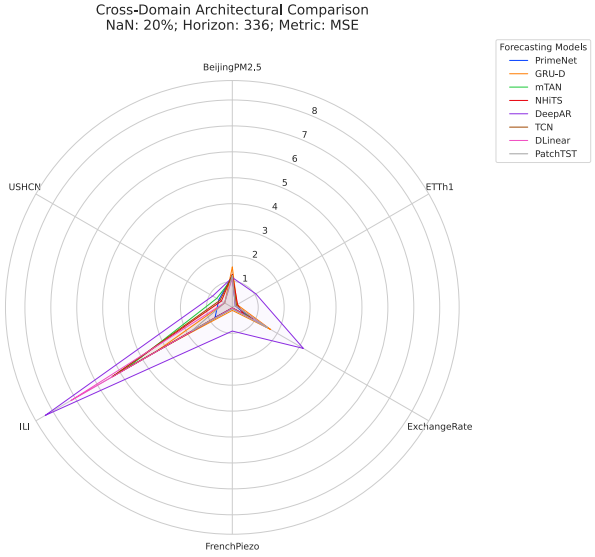


(c)

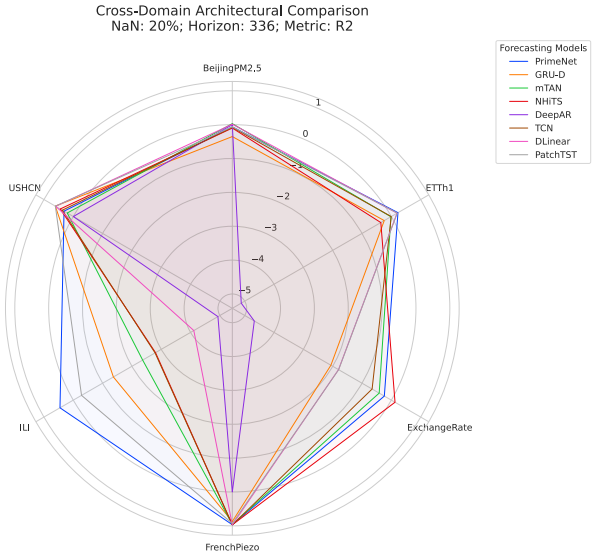
Figure A.6: Radar plot over MAE, MSE and R^2 Score metrics averaging over 20% missing data and a forecasting horizon of 192 (36 for ILI and USHCN).



(a)

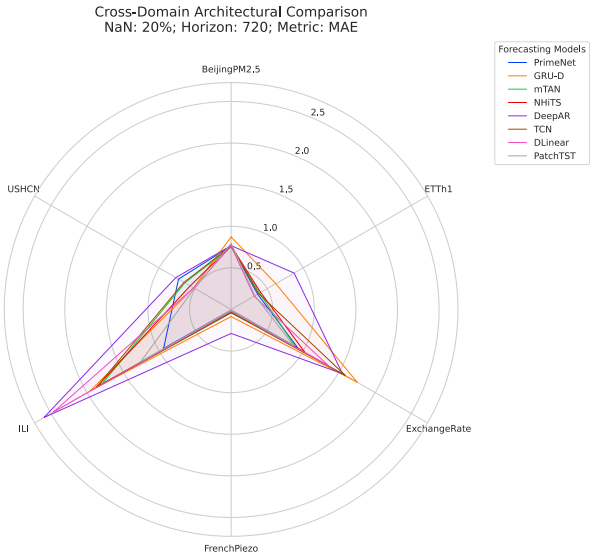


(b)

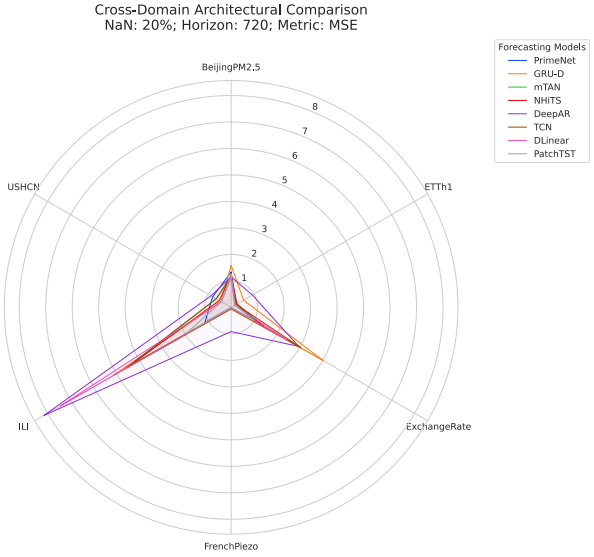


(c)

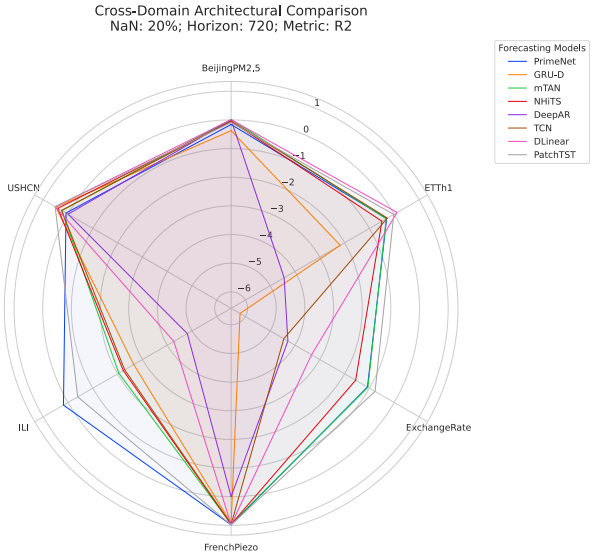
Figure A.7: Radar plot over MAE, MSE and R^2 Score metrics averaging over 20% missing data and a forecasting horizon of 336 (48 for ILI and USHCN).



(a)

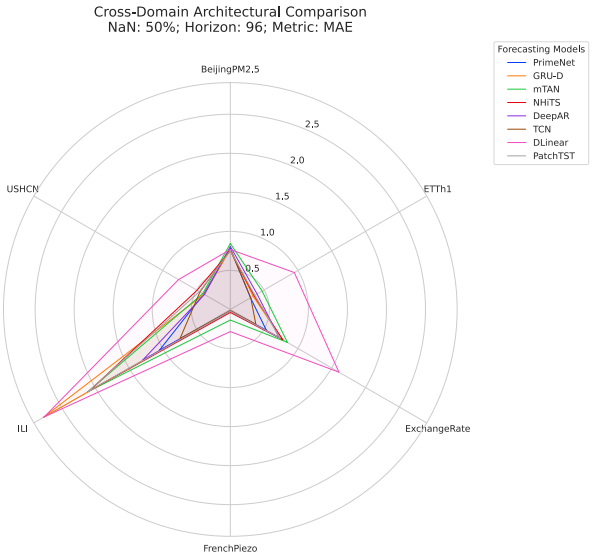


(b)

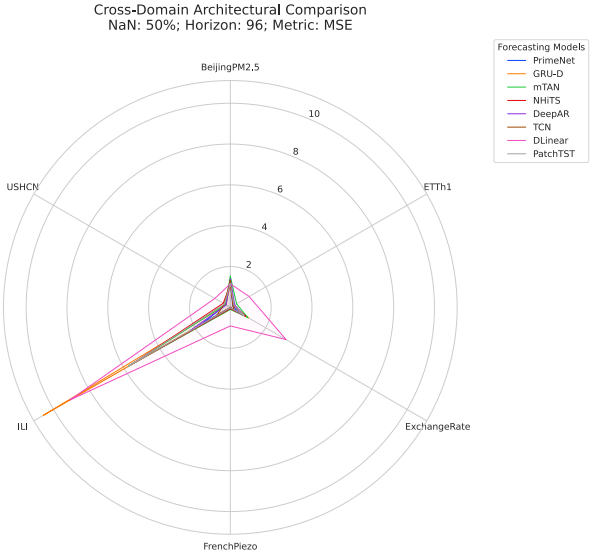


(c)

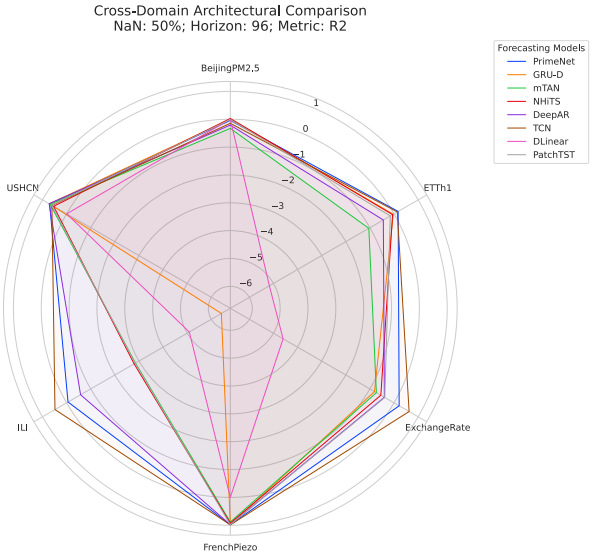
Figure A.8: Radar plot over MAE, MSE and R^2 Score metrics averaging over 20% missing data and a forecasting horizon of 720 (60 for ILI and USHCN).



(a)

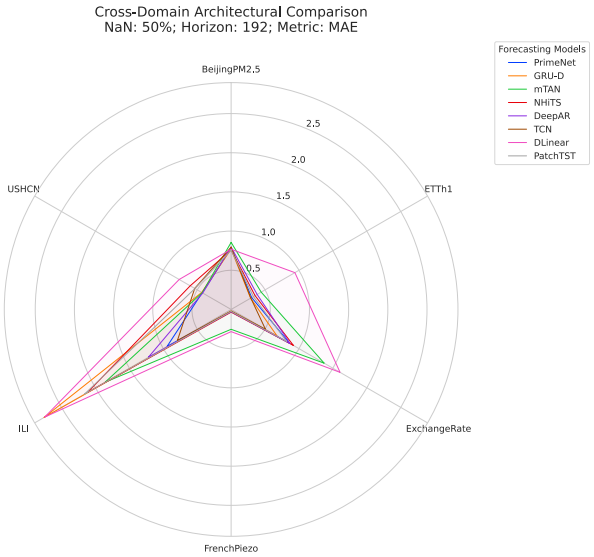


(b)

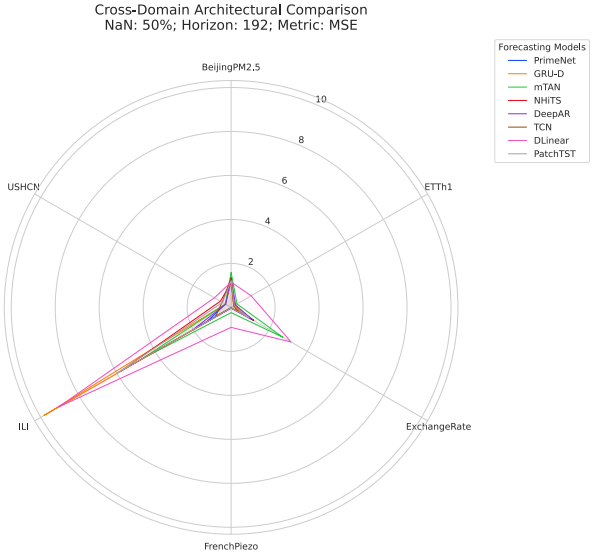


(c)

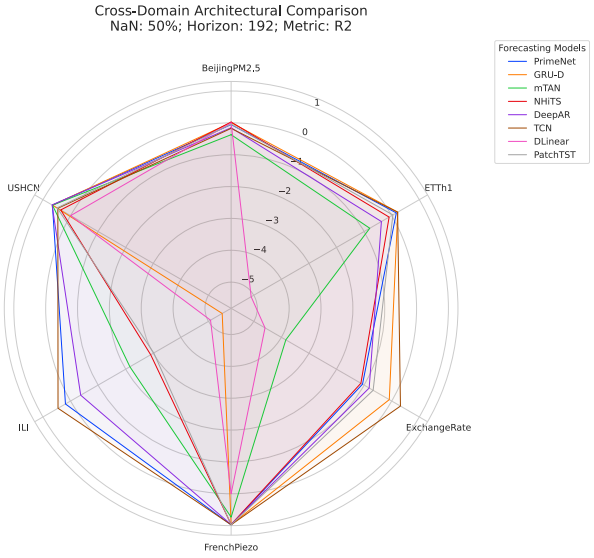
Figure A.9: Radar plot over MAE, MSE and R^2 Score metrics averaging over 50% missing data and a forecasting horizon of 96 (24 for ILI and USHCN).



(a)

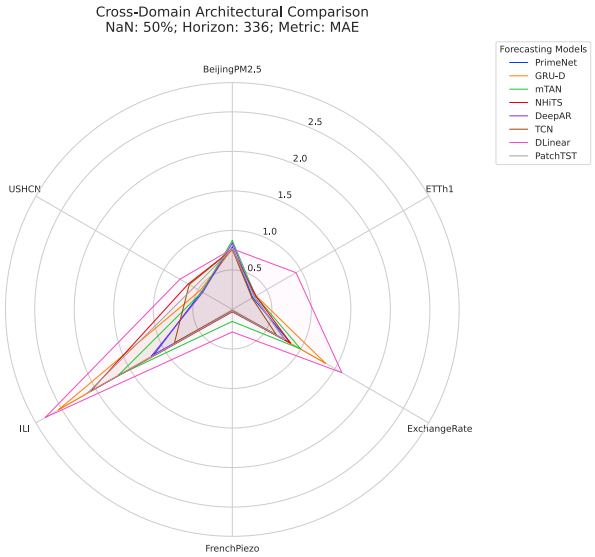


(b)

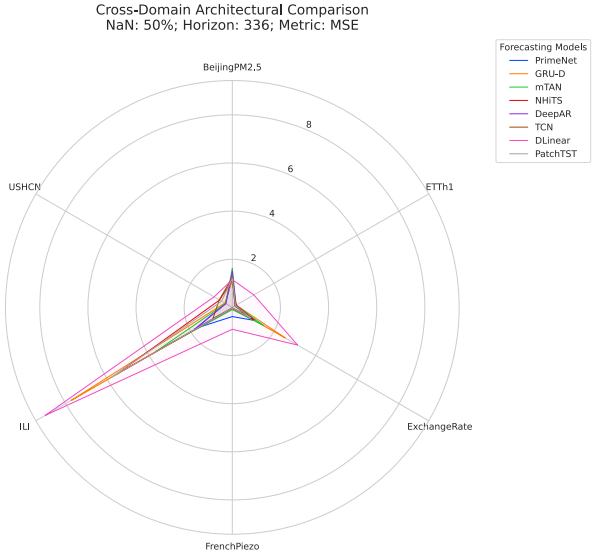


(c)

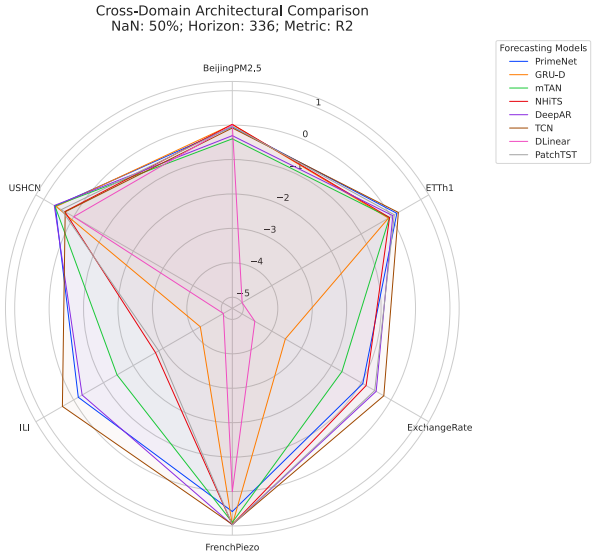
Figure A.10: Radar plot over MAE, MSE and R^2 Score metrics averaging over 50% missing data and a forecasting horizon of 192 (36 for ILI and USHCN).



(a)

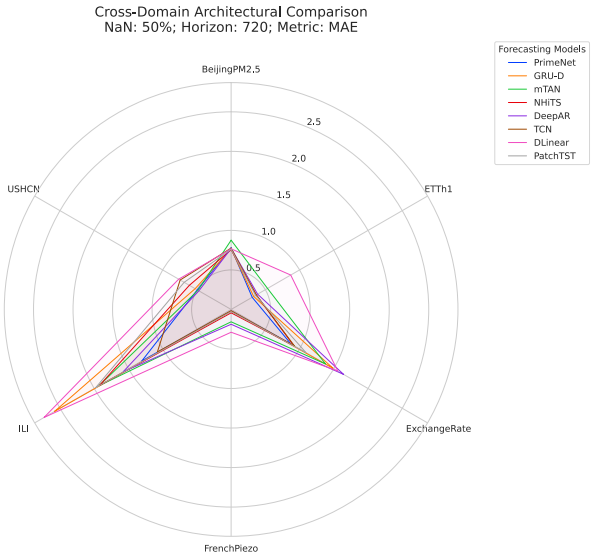


(b)

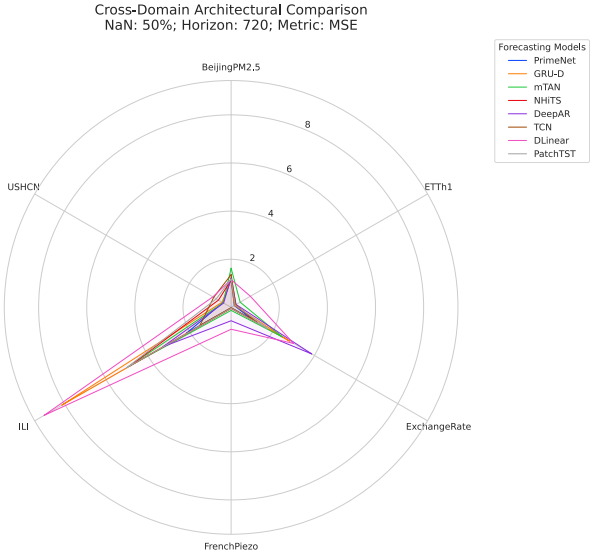


(c)

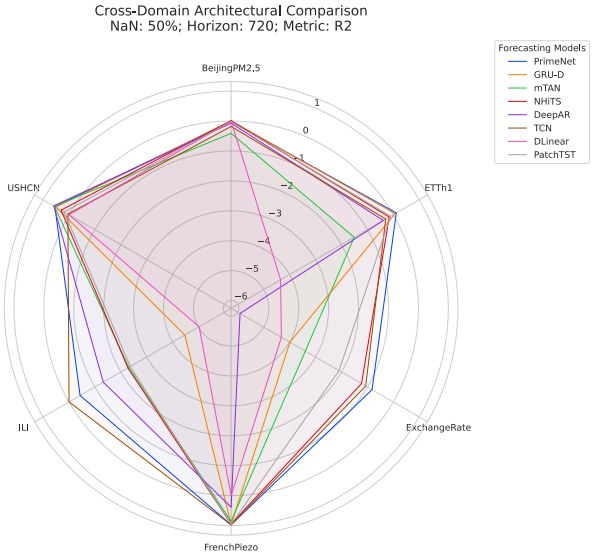
Figure A.11: Radar plot over MAE, MSE and R^2 Score metrics averaging over 50% missing data and a forecasting horizon of 336 (48 for ILI and USHCN).



(a)

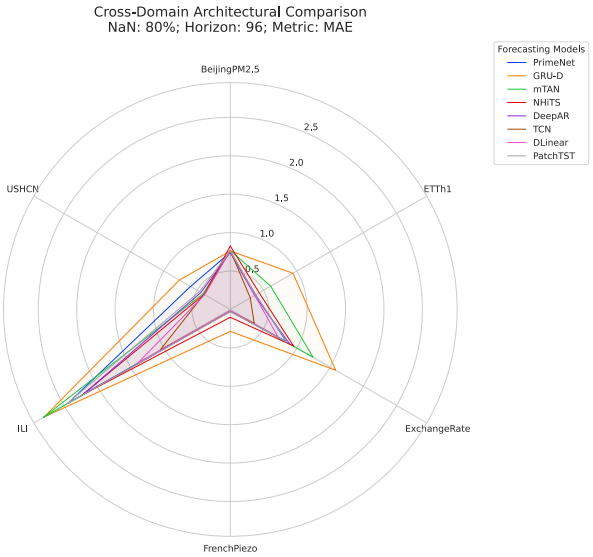


(b)

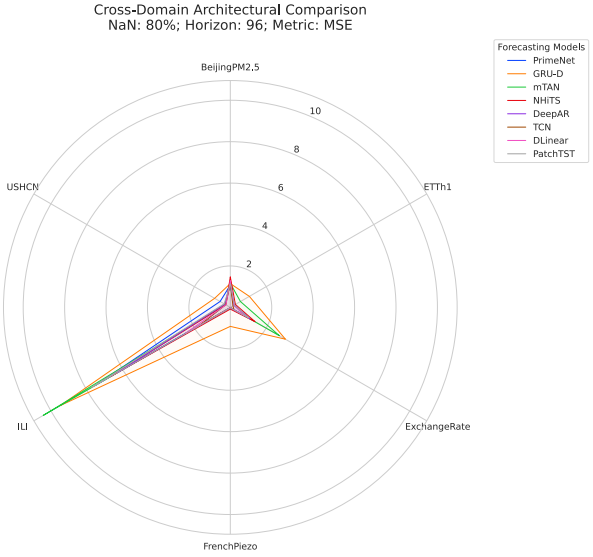


(c)

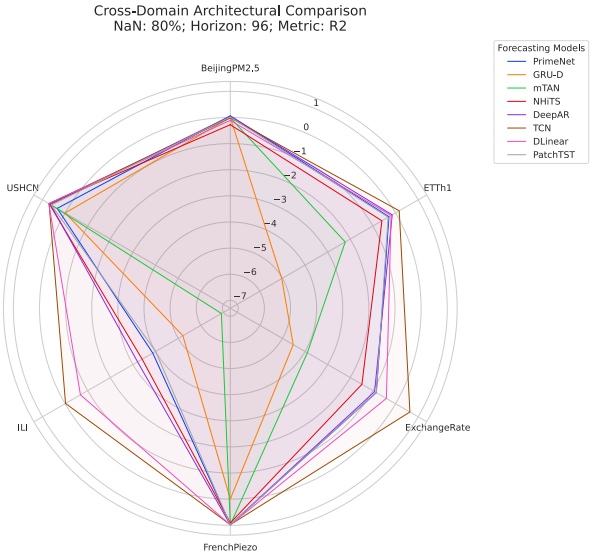
Figure A.12: Radar plot over MAE, MSE and R^2 Score metrics averaging over 50% missing data and a forecasting horizon of 720 (60 for ILI and USHCN).



(a)

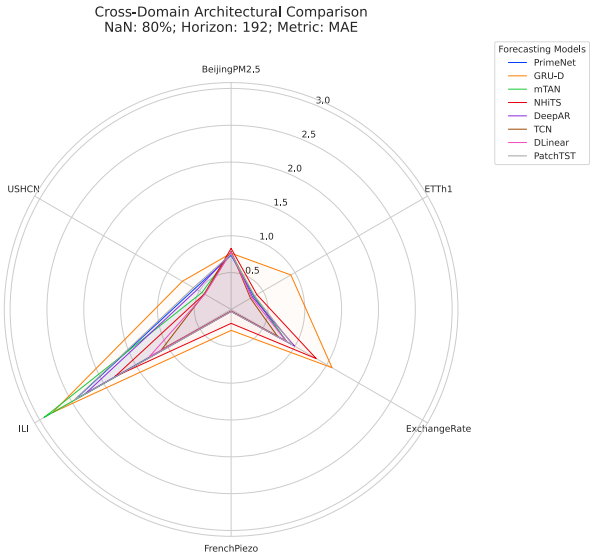


(b)

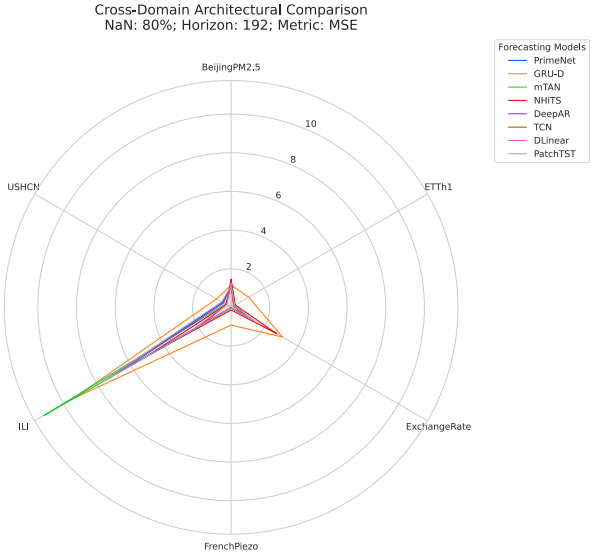


(c)

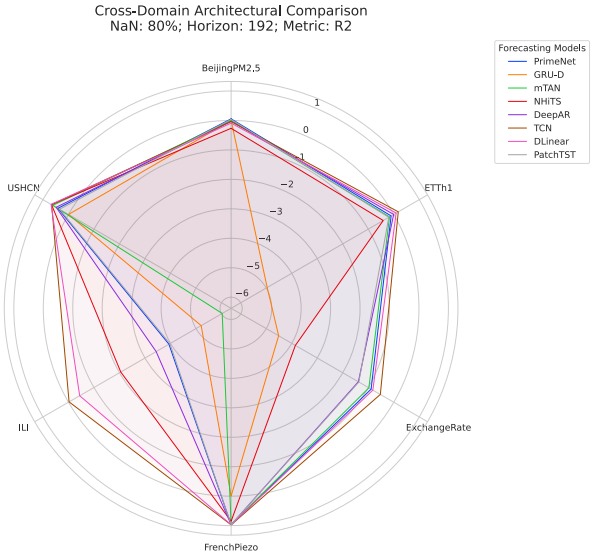
Figure A.13: Radar plot over MAE, MSE and R^2 Score metrics averaging over 80% missing data and a forecasting horizon of 96 (60 for ILI and USHCN).



(a)

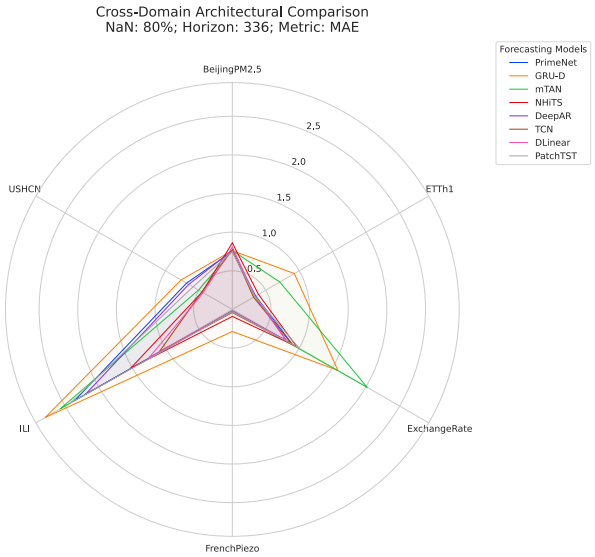


(b)

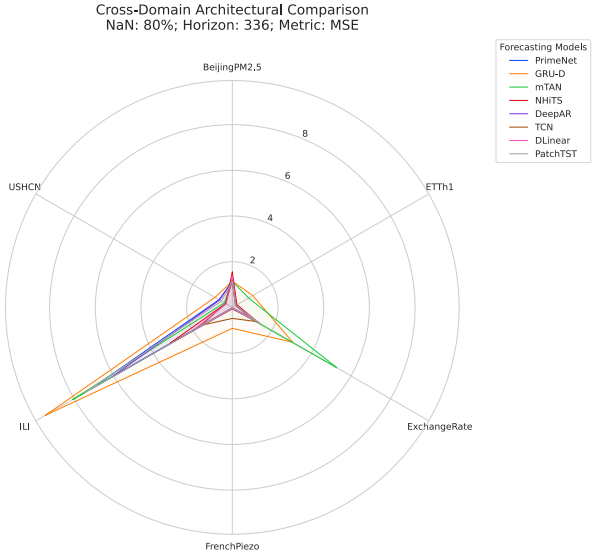


(c)

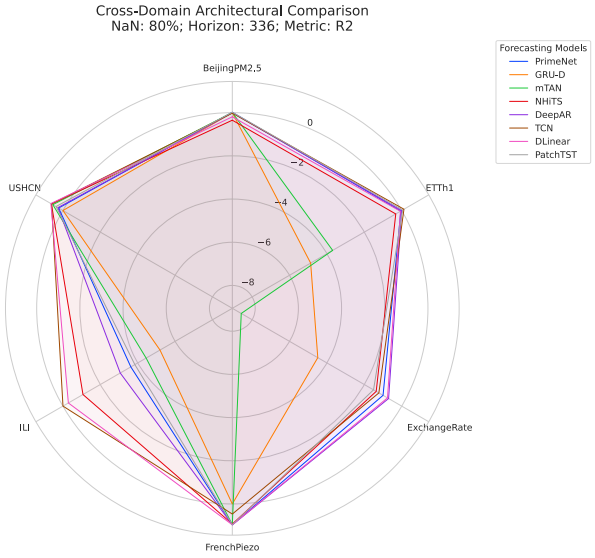
Figure A.14: Radar plot over MAE, MSE and R^2 Score metrics averaging over 80% missing data and a forecasting horizon of 192 (36 for ILI and USHCN).



(a)

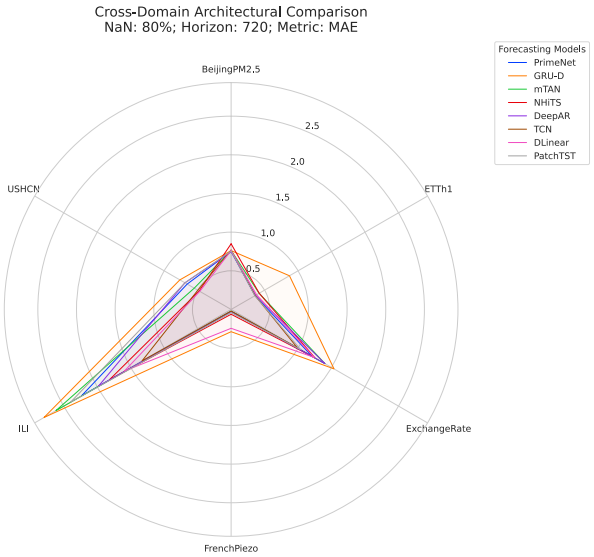


(b)

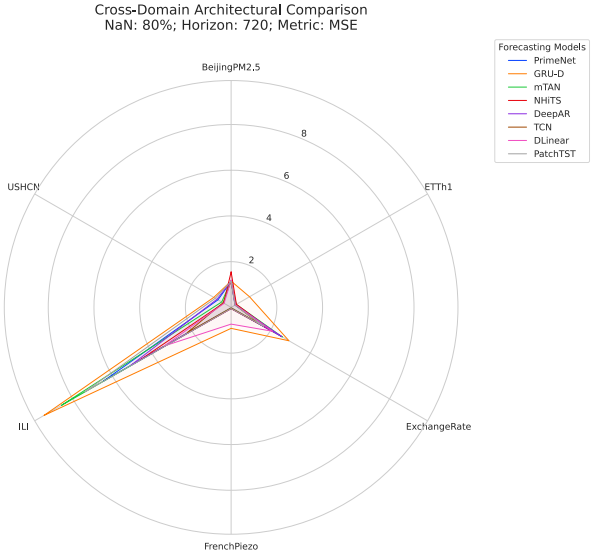


(c)

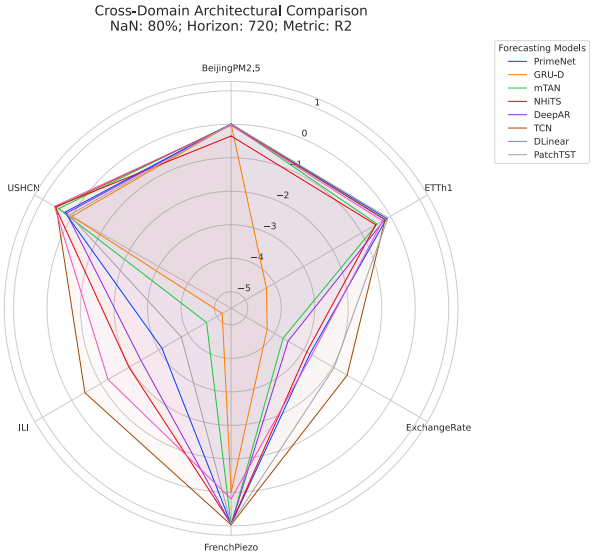
Figure A.15: Radar plot over MAE, MSE and R^2 Score metrics averaging over 80% missing data and a forecasting horizon of 336 (48 for ILI and USHCN).



(a)



(b)



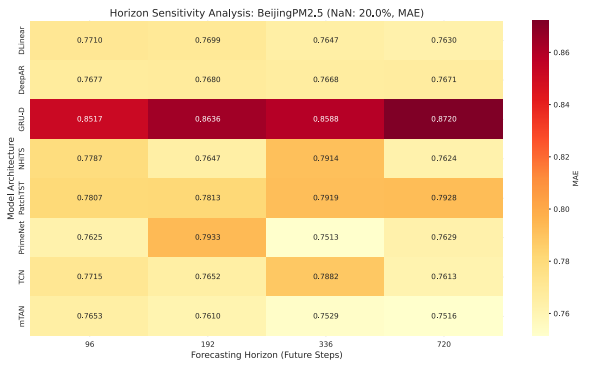
(c)

Figure A.16: Radar plot over MAE, MSE and R^2 Score metrics averaging over 80% missing data and a forecasting horizon of 720 (60 for ILI and USHCN).

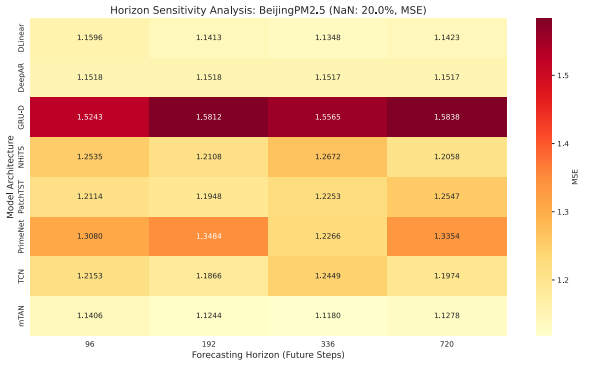
A.3.3 Horizon Sensitivity Heatmaps

Heatmaps illustrate the degradation of MAE and MSE metrics as the forecasting horizon increases ($H \in \{96, 192, 336, 720\}$. Each line of the heatmap shows the performance of a model over the horizons. The colormap was normalized so that it's distributed per column; this way, the colors reflect the performance of the models on that horizon.

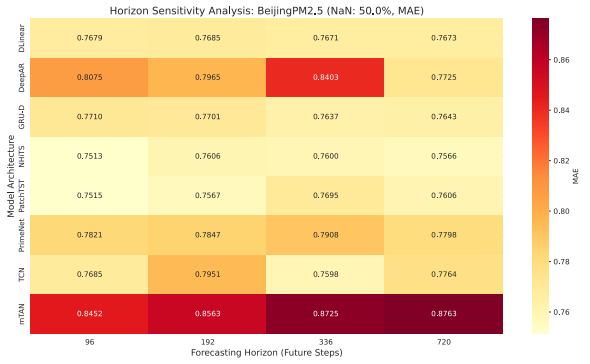
One page per dataset, showing the 3 NaN% levels.



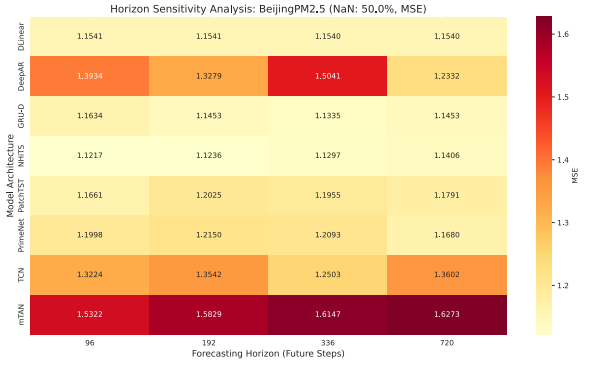
(a)



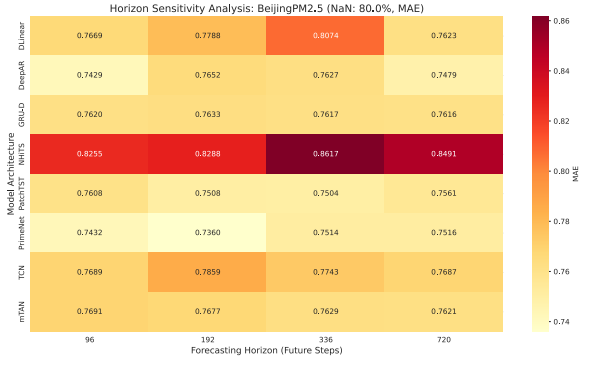
(b)



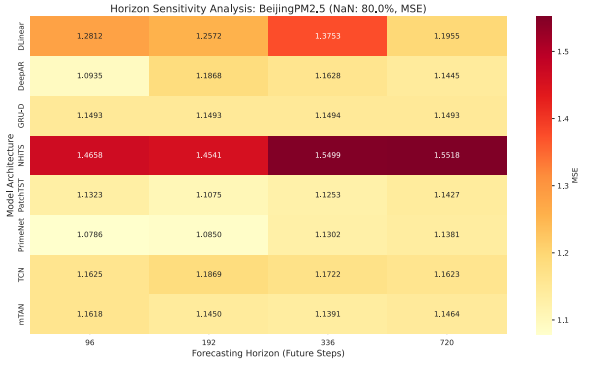
(c)



(d)

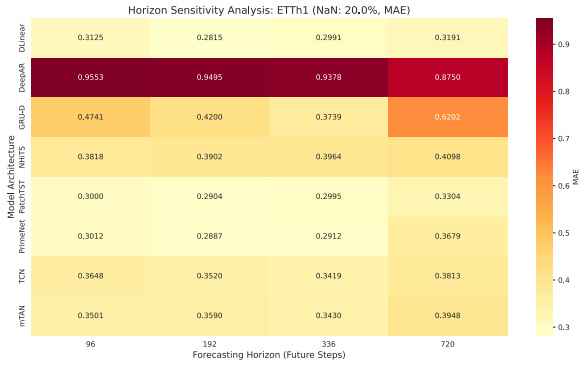


(e)

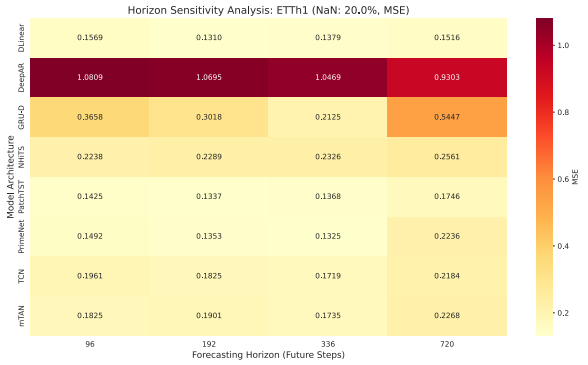


(f)

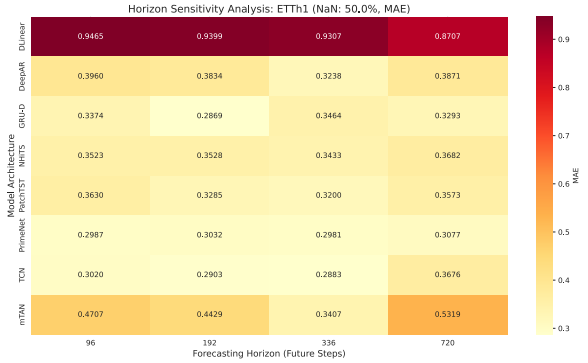
Figure A.17: Horizon Sensitivity Heatmaps on BeijingPM2.5 across all forecasting horizons.



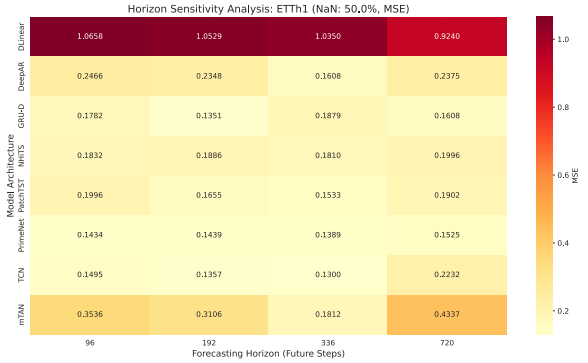
(a)



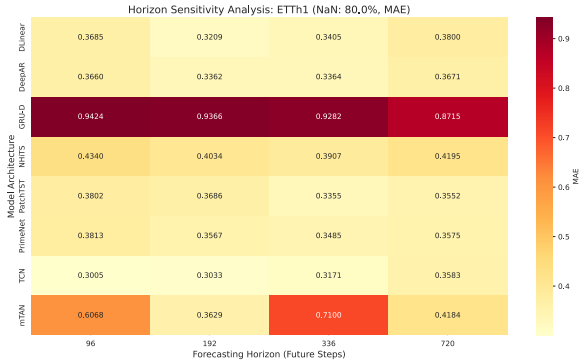
(b)



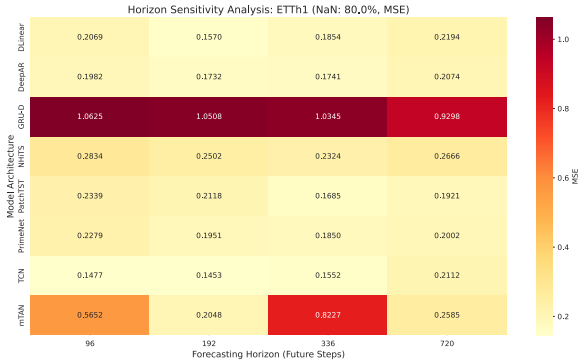
(c)



(d)

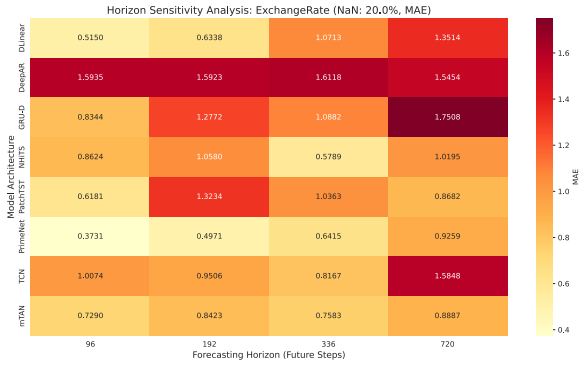


(e)

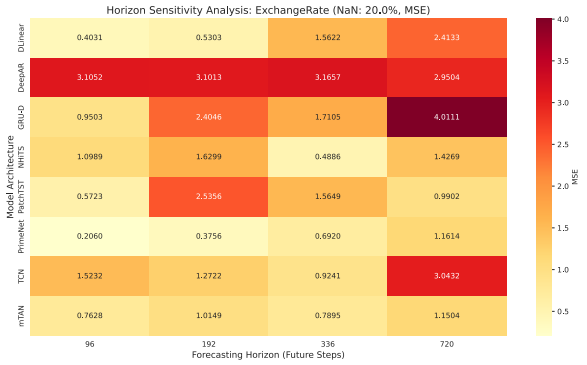


(f)

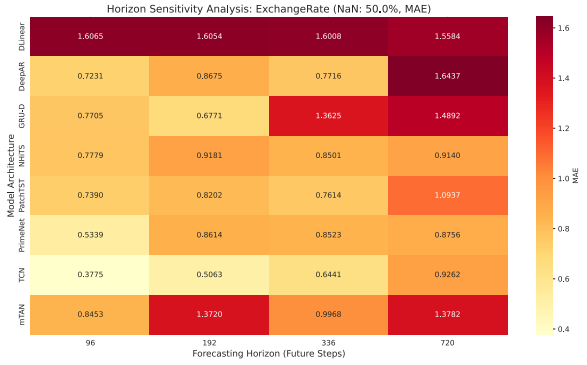
Figure A.18: Horizon Sensitivity Heatmaps on ETTh1 across all forecasting horizons.



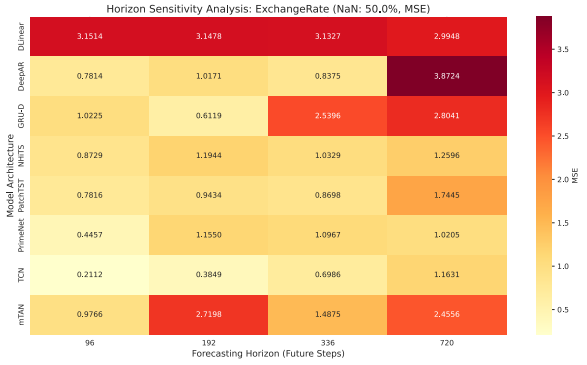
(a)



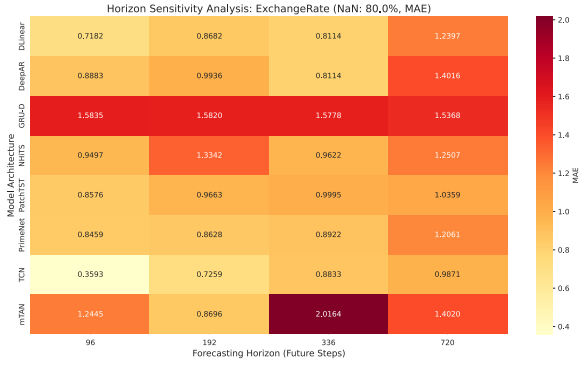
(b)



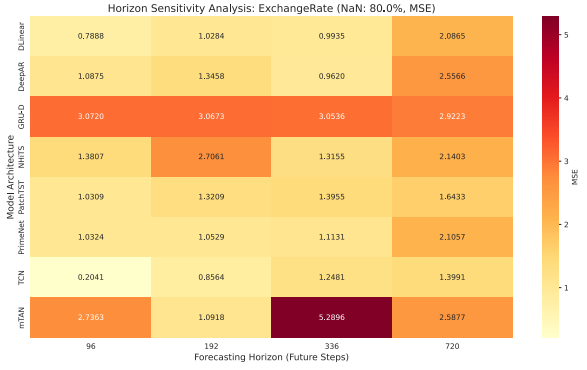
(c)



(d)

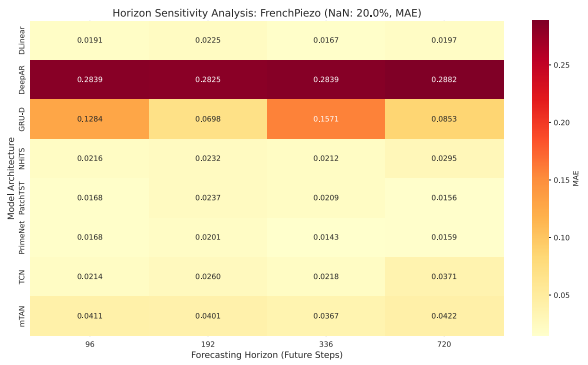


(e)

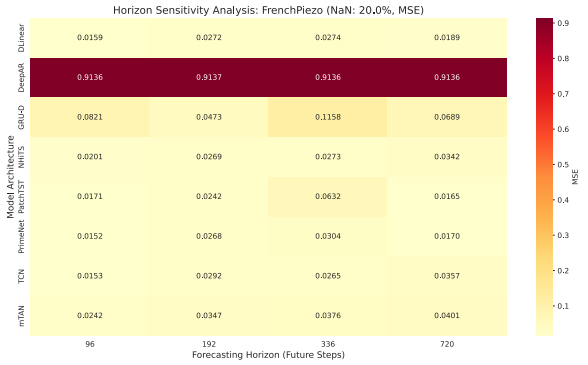


(f)

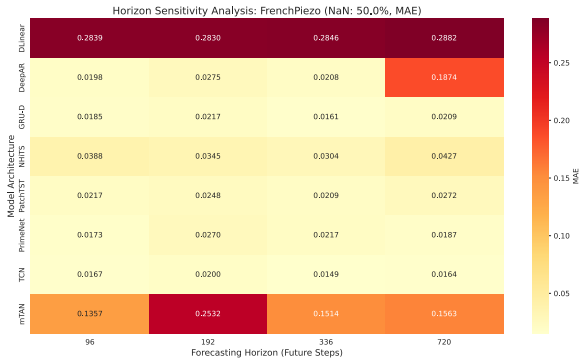
Figure A.19: Horizon Sensitivity Heatmaps on ExchangeRate across all forecasting horizons.



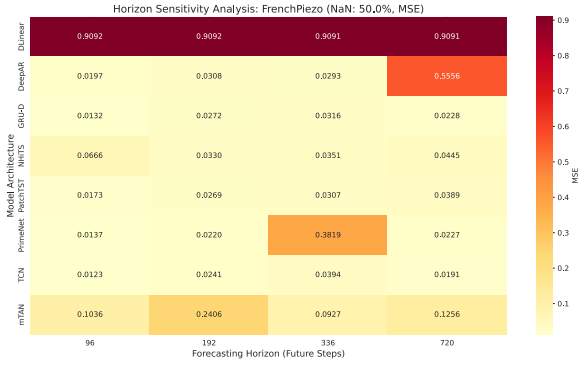
(a)



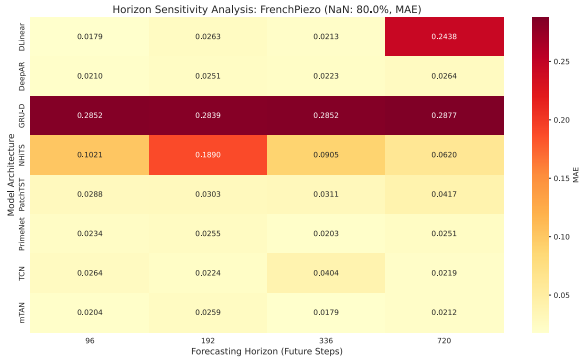
(b)



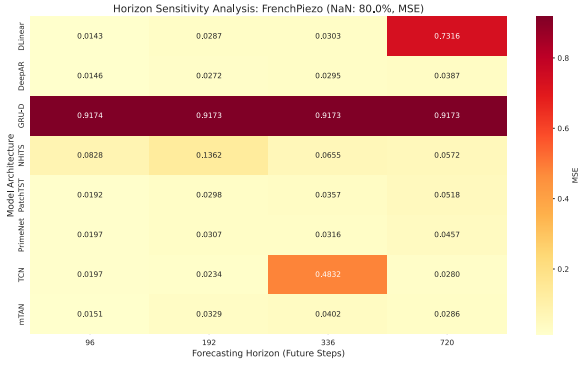
(c)



(d)

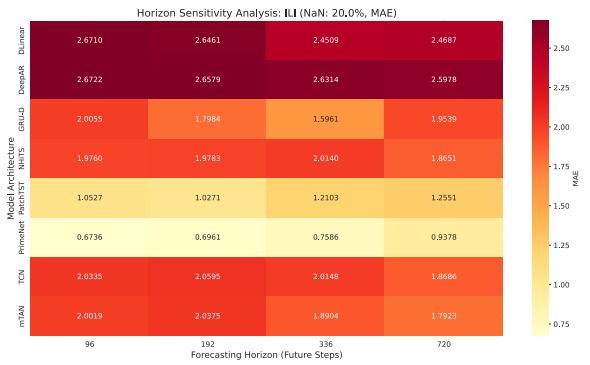


(e)

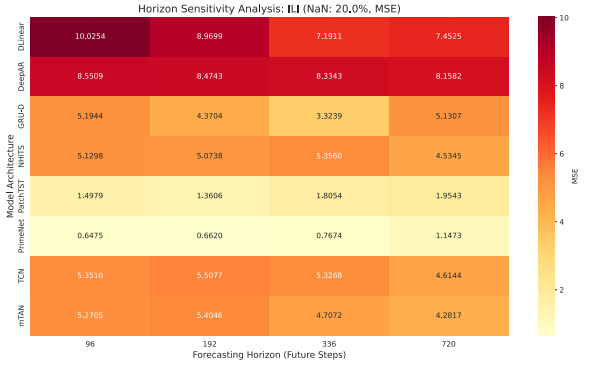


(f)

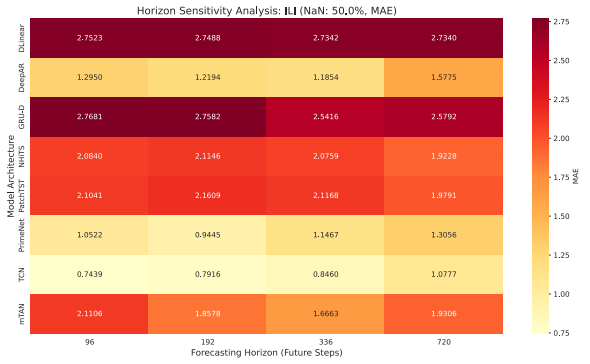
Figure A.20: Horizon Sensitivity Heatmaps on FrenchPiezo across all forecasting horizons.



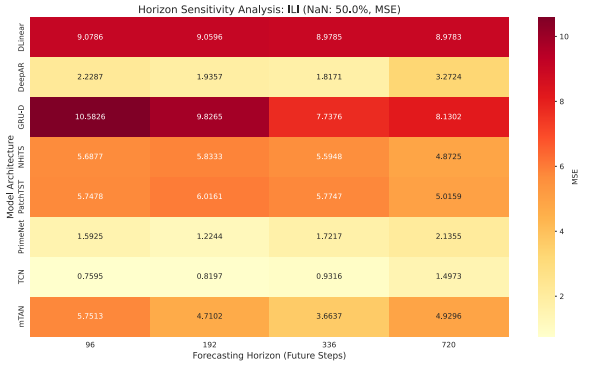
(a)



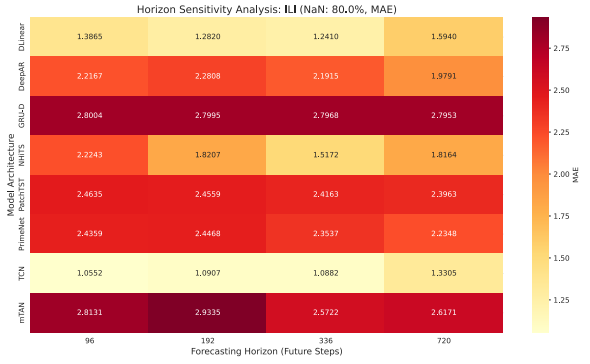
(b)



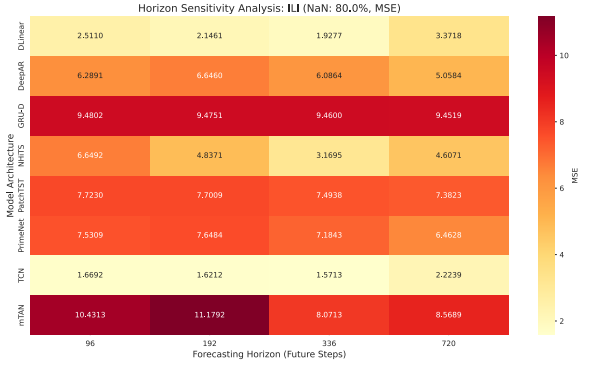
(c)



(d)

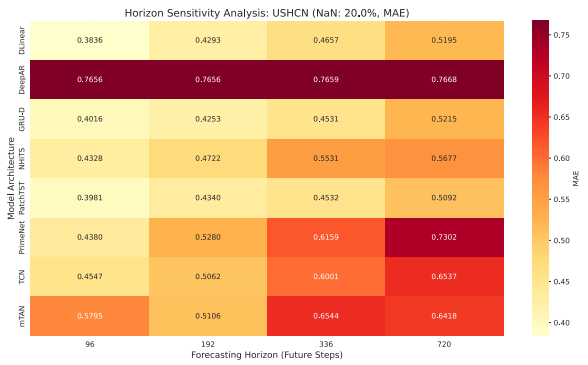


(e)

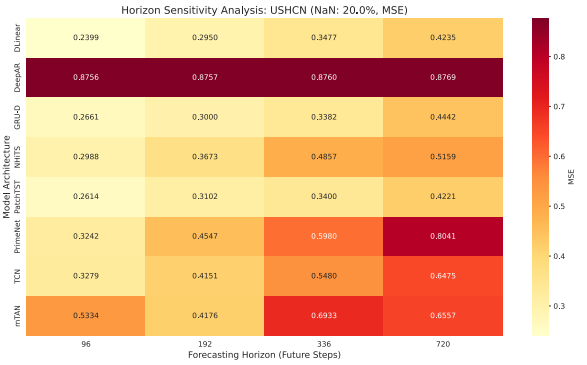


(f)

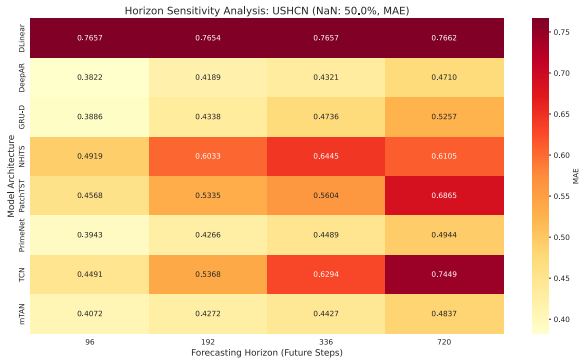
Figure A.21: Horizon Sensitivity Heatmaps on ILI across all forecasting horizons.



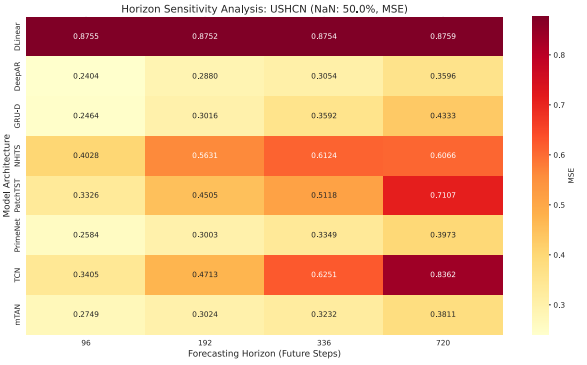
(a)



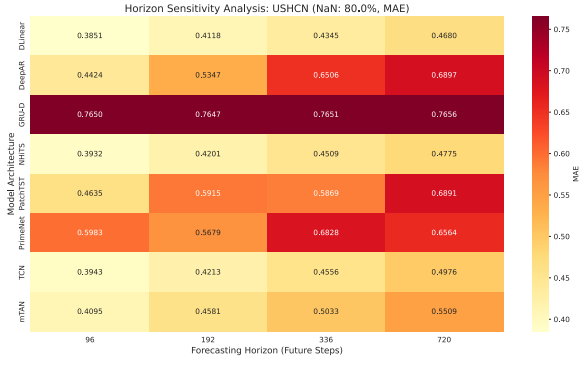
(b)



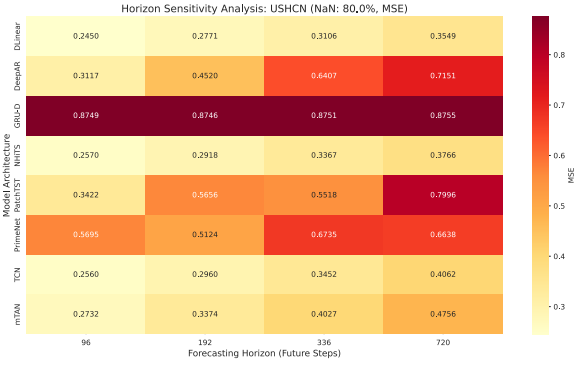
(c)



(d)



(e)

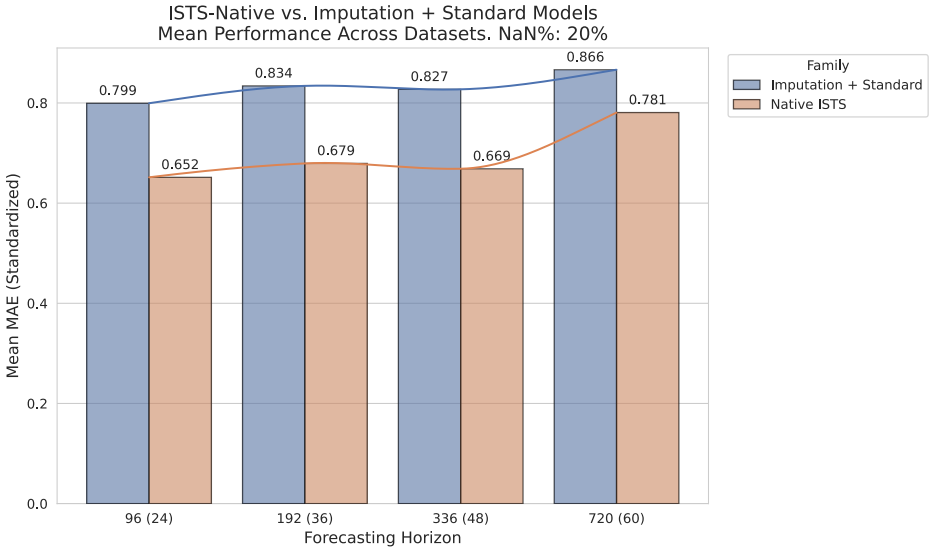


(f)

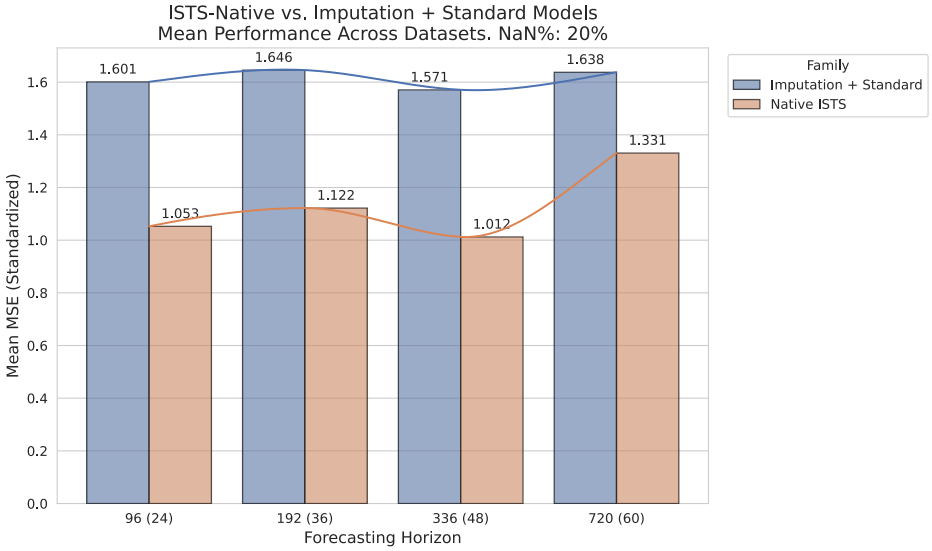
Figure A.22: Horizon Sensitivity Heatmaps on USHCN across all forecasting horizons.

A.3.4 Stochastic Robustness Barplots

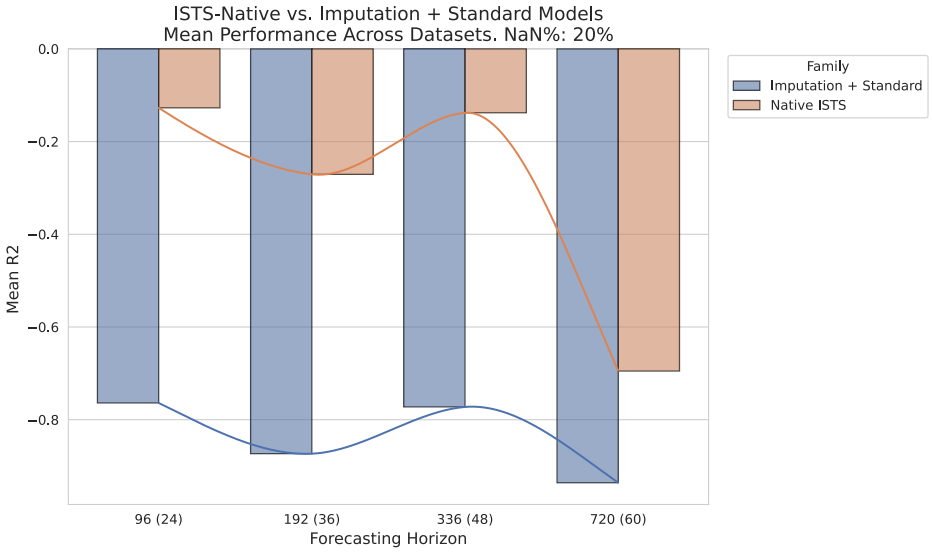
Bar charts show the mean performance across five randomness seeds ($0 - 4$), with a smoothed line over each bar to highlight the trend of architectural sensitivity to specific missingness patterns between the Native ISTS and Imputation + Standard model families.



(a)

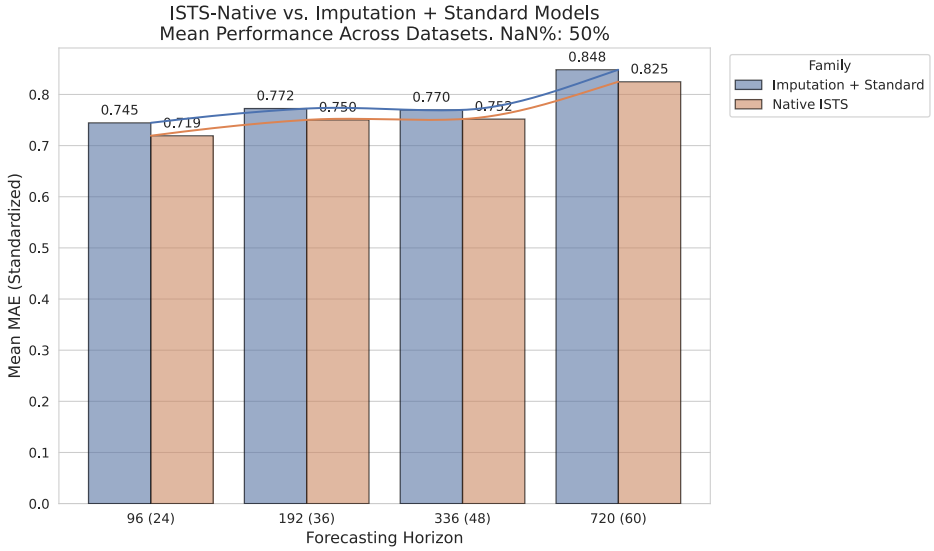


(b)

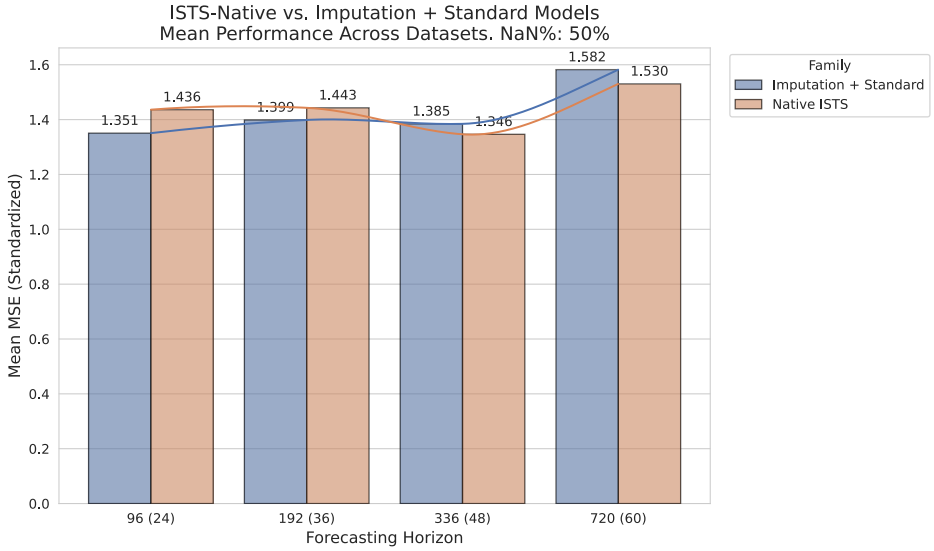


(c)

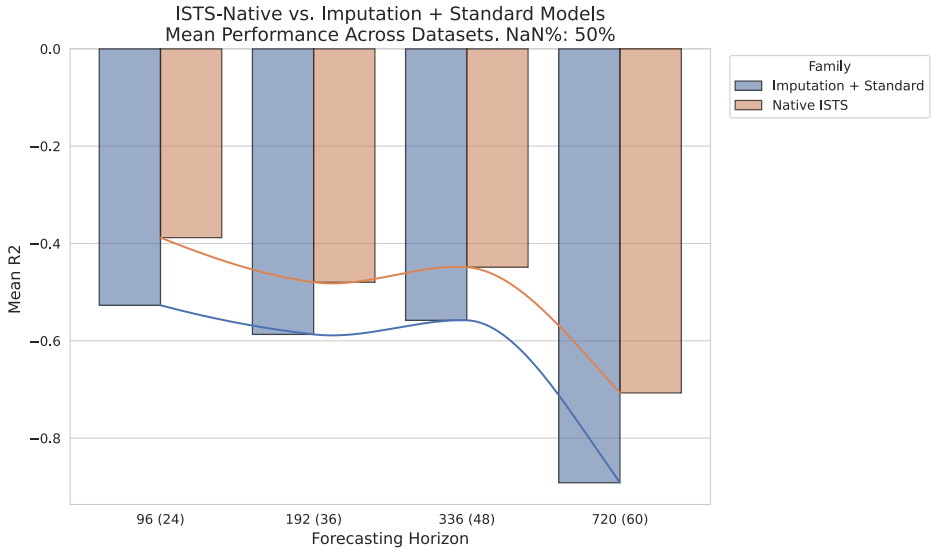
Figure A.23: Grouped barplots showing Native ISTS models performance vs Imputation + Standard model pipelines on all datasets with 20% missing data.



(a)

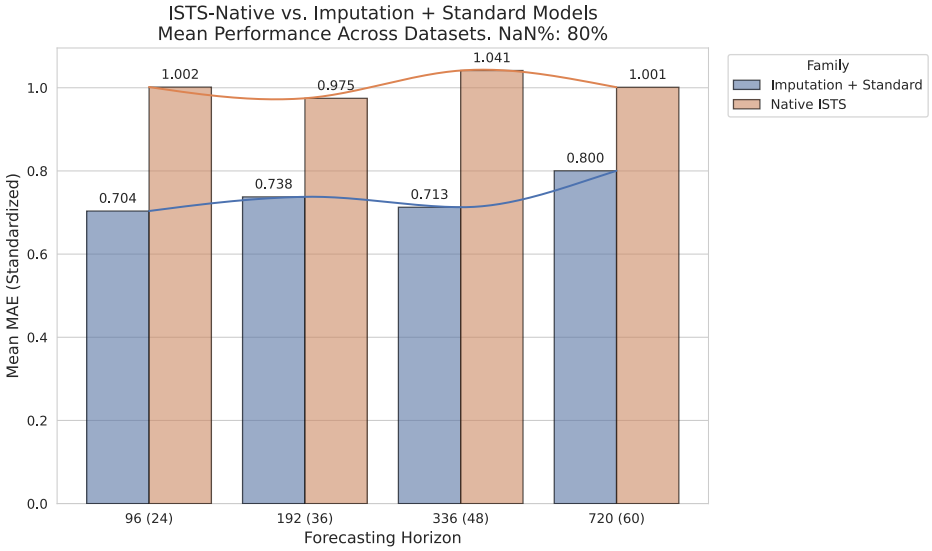


(b)

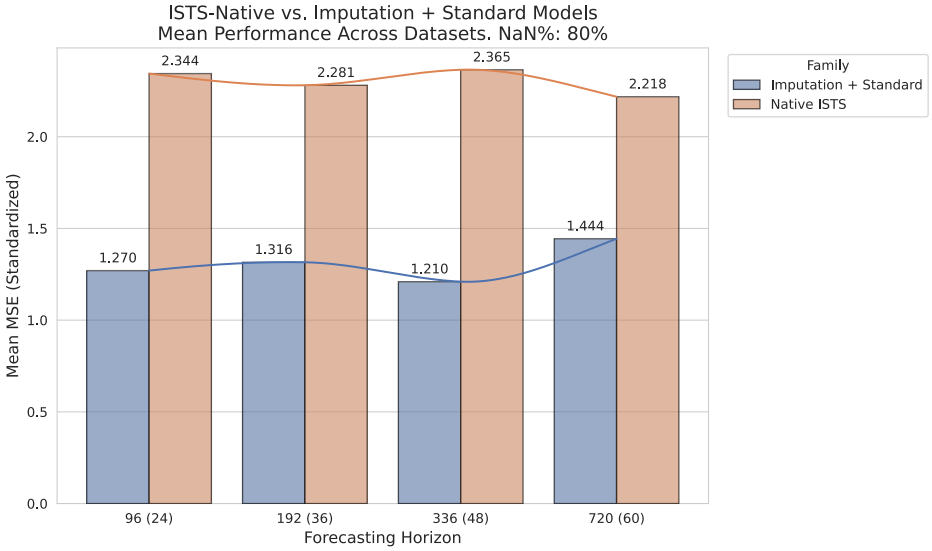


(c)

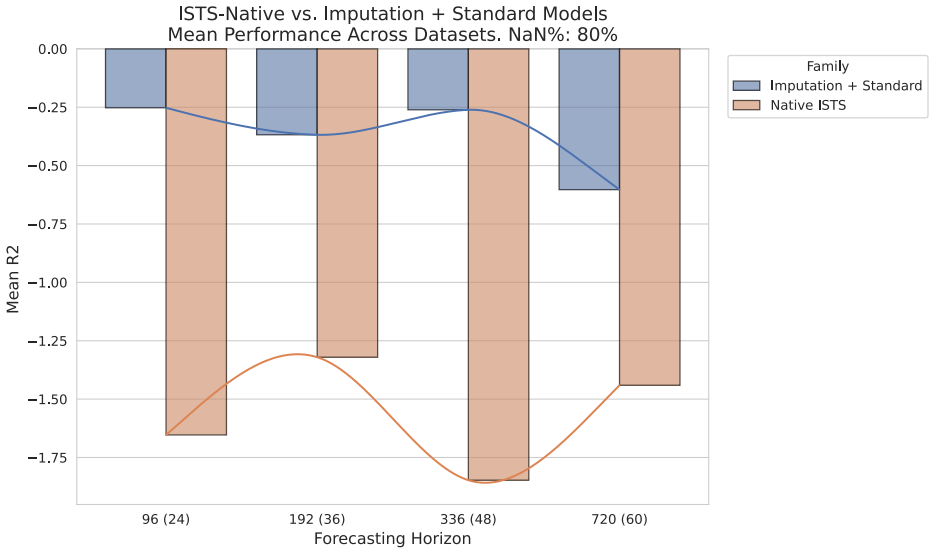
Figure A.24: Grouped barplots showing Native ISTS models performance vs Imputation + Standard model pipelines on all datasets with 50% missing data.



(a)

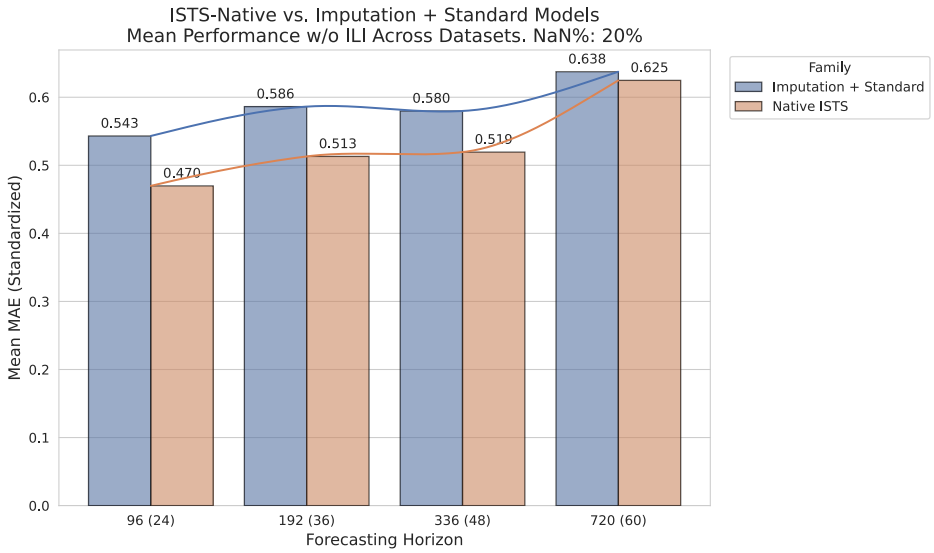


(b)

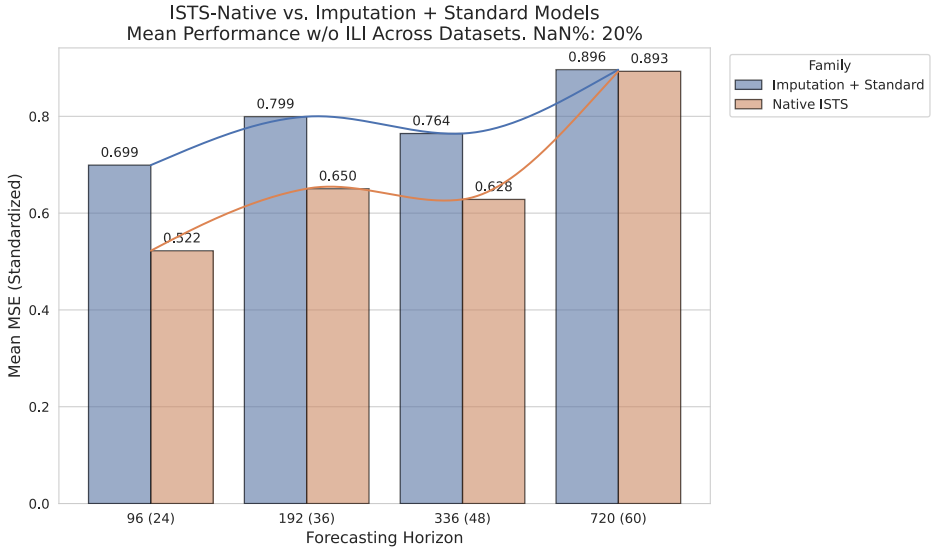


(c)

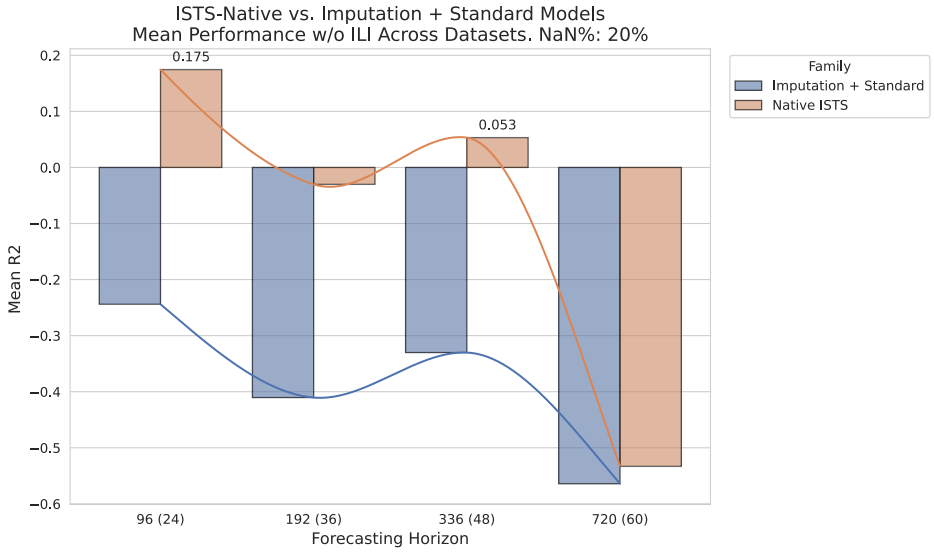
Figure A.25: Grouped barplots showing Native ISTS models performance vs Imputation + Standard model pipelines on all datasets with 80% missing data.



(a)

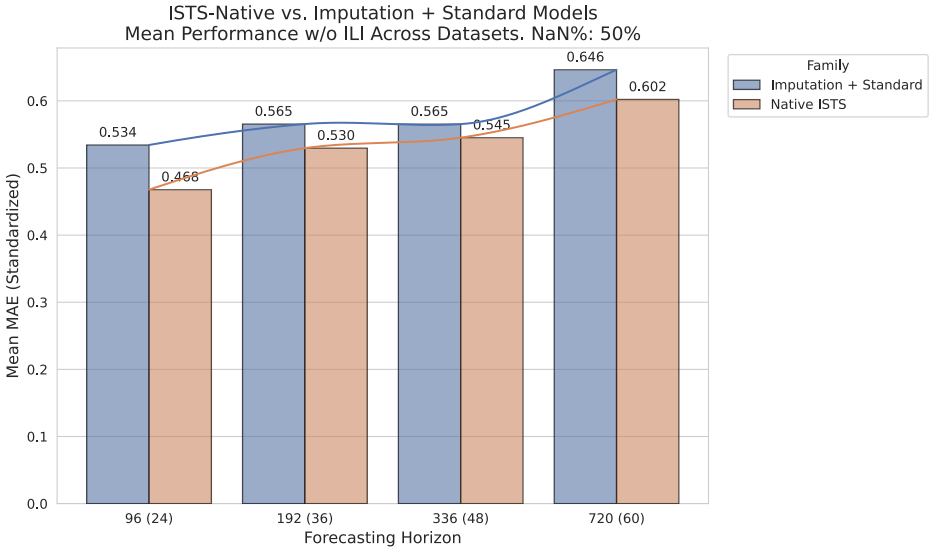


(b)

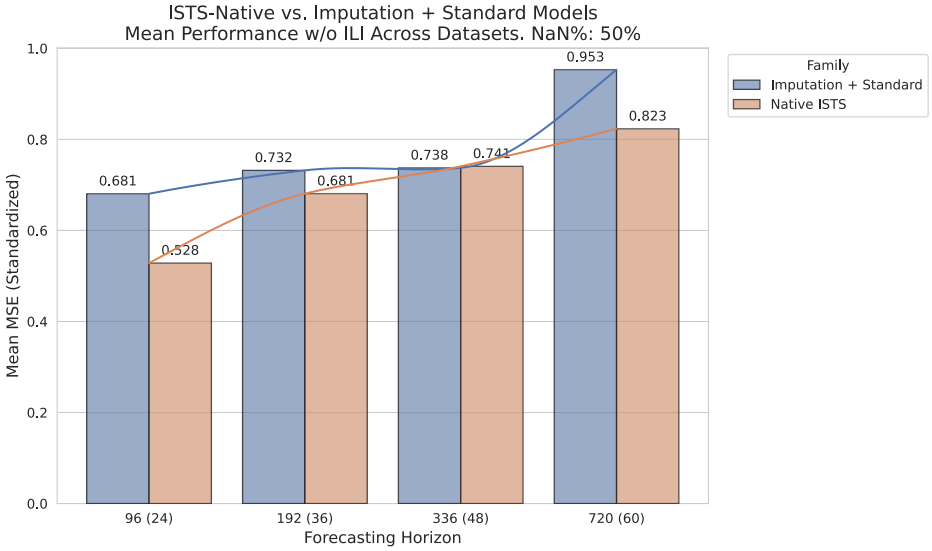


(c)

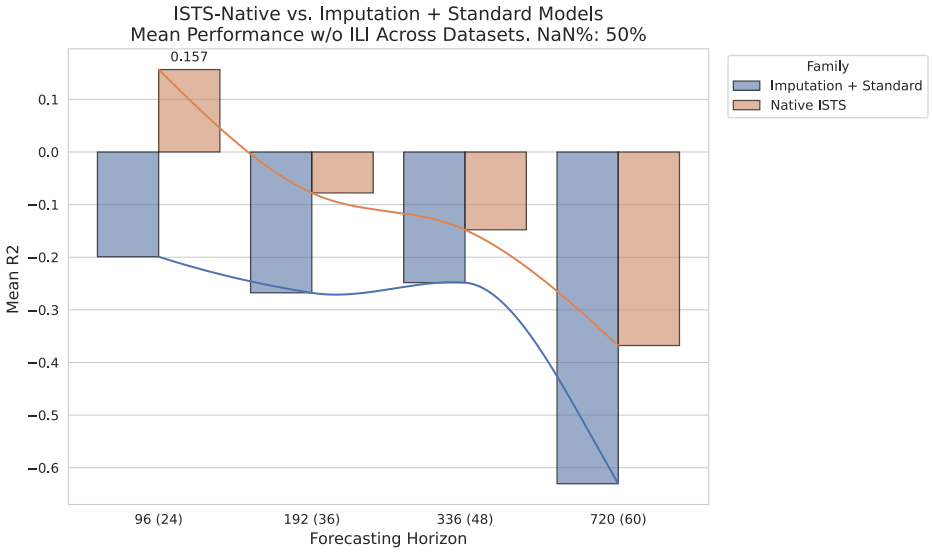
Figure A.26: Grouped barplots showing Native ISTS models performance vs Imputation + Standard model pipelines on all datasets **excluding** ILI with 20% missing data.



(a)

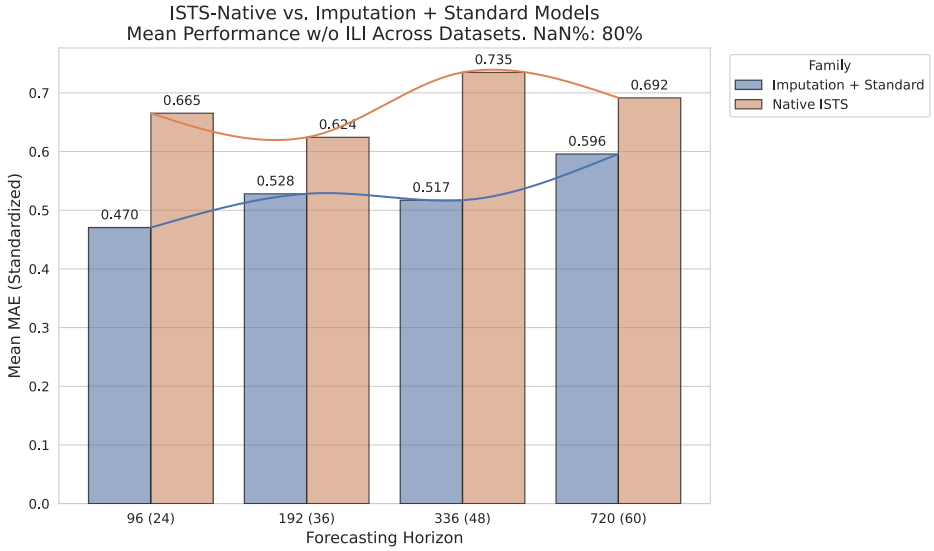


(b)

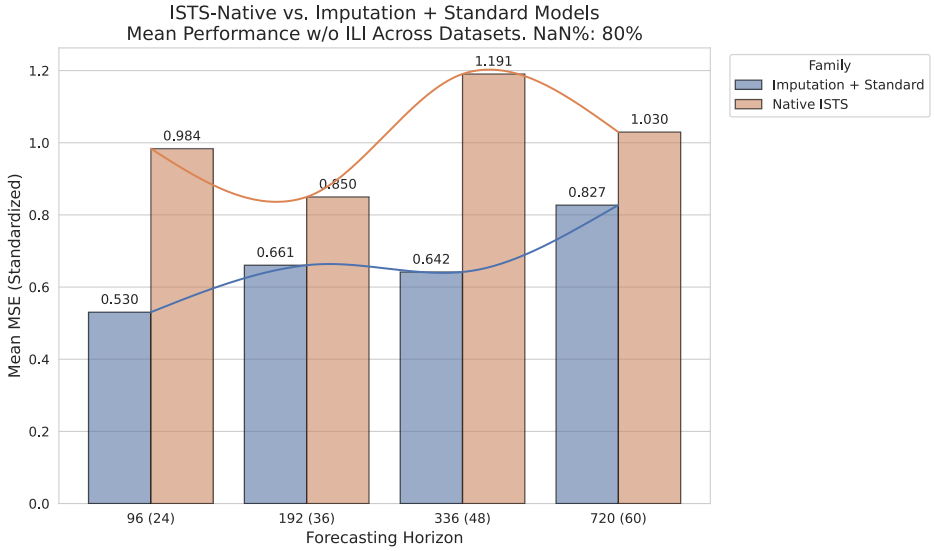


(c)

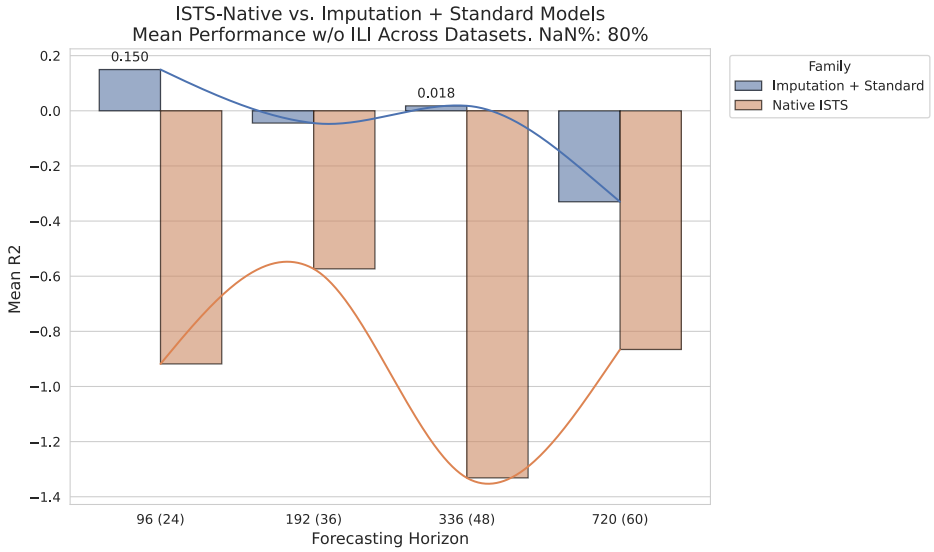
Figure A.27: Grouped barplots showing Native ISTS models performance vs Imputation + Standard model pipelines on all datasets **excluding** ILI with 50% missing data.



(a)

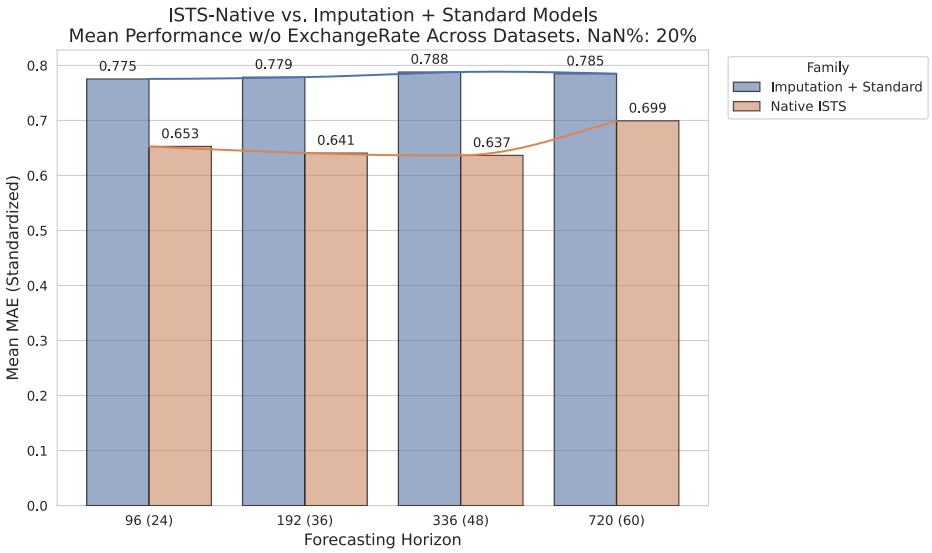


(b)

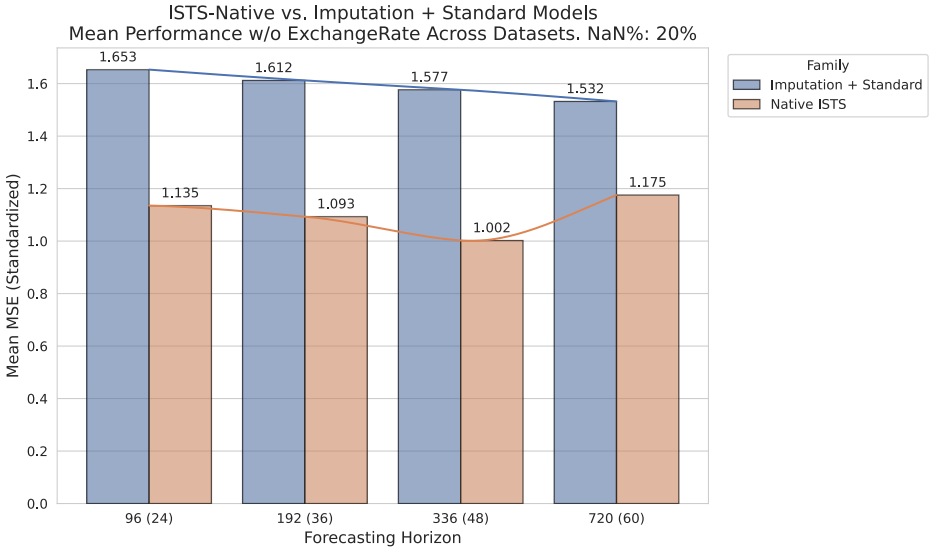


(c)

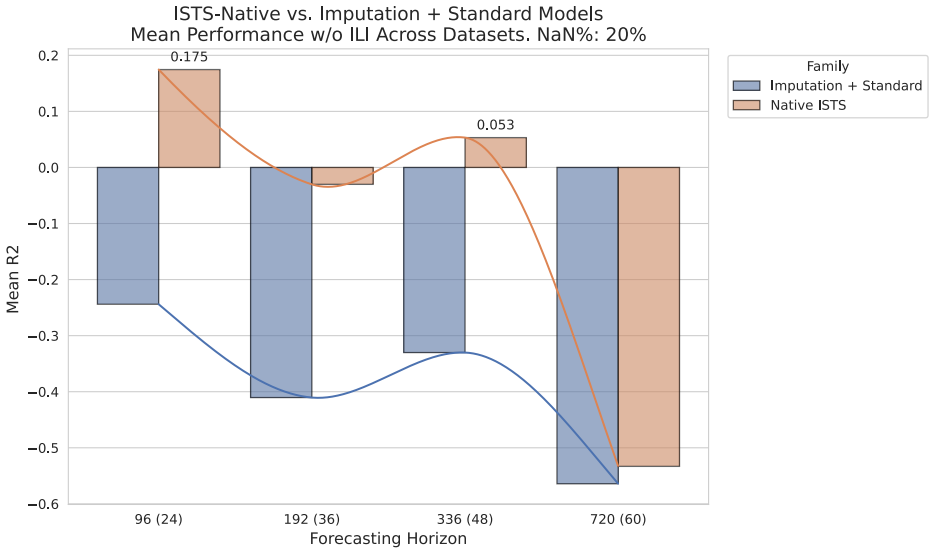
Figure A.28: Grouped barplots showing Native ISTS models performance vs Imputation + Standard model pipelines on all datasets **excluding** ILI with 80% missing data.



(a)

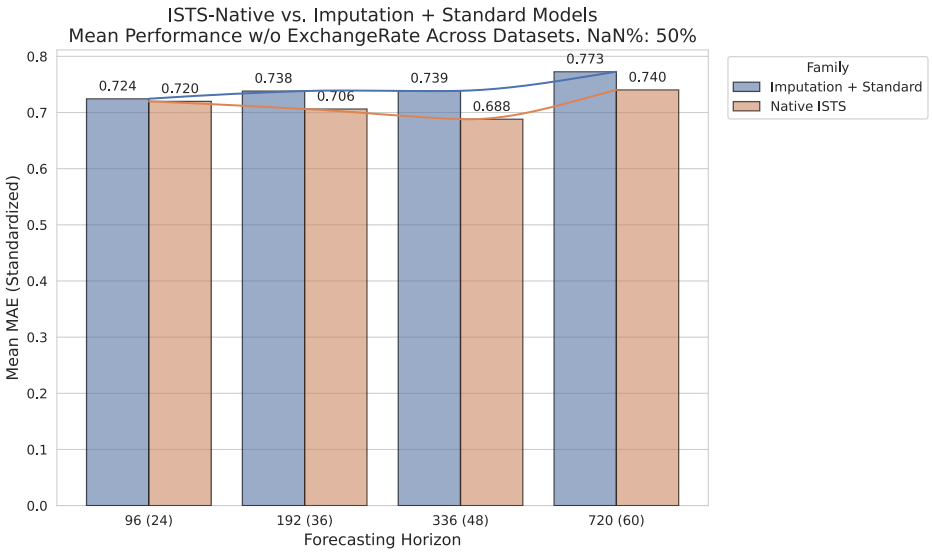


(b)

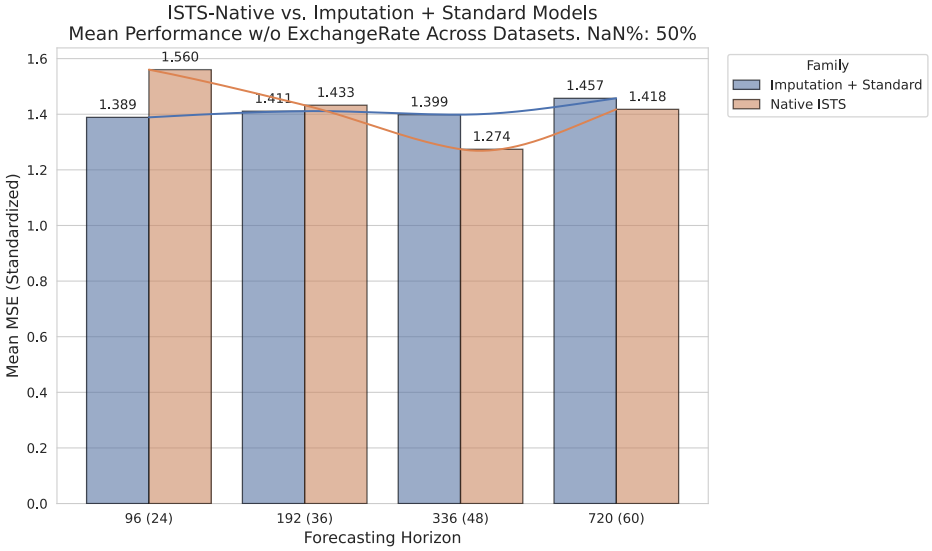


(c)

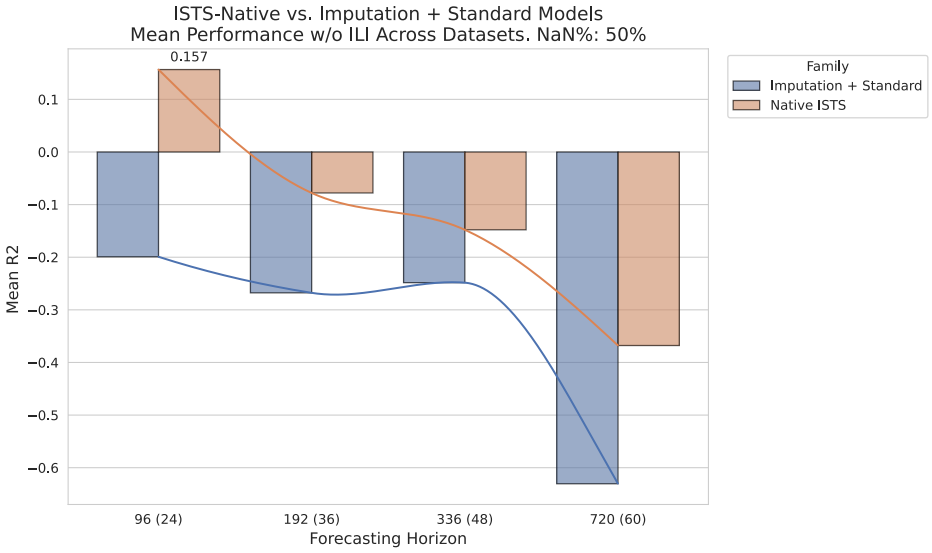
Figure A.29: Grouped barplots showing Native ISTS models performance vs Imputation + Standard model pipelines on all datasets **excluding** ExchangeRate with 20% missing data.



(a)

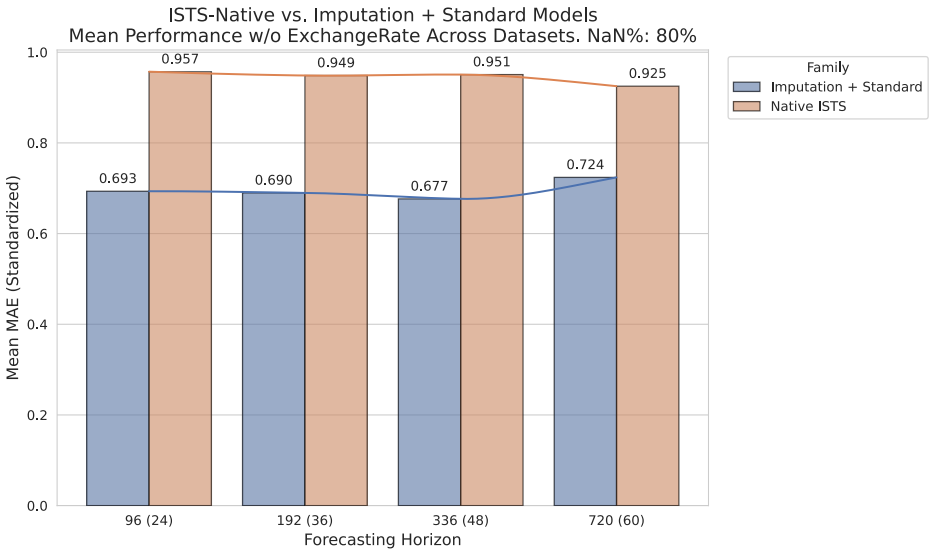


(b)

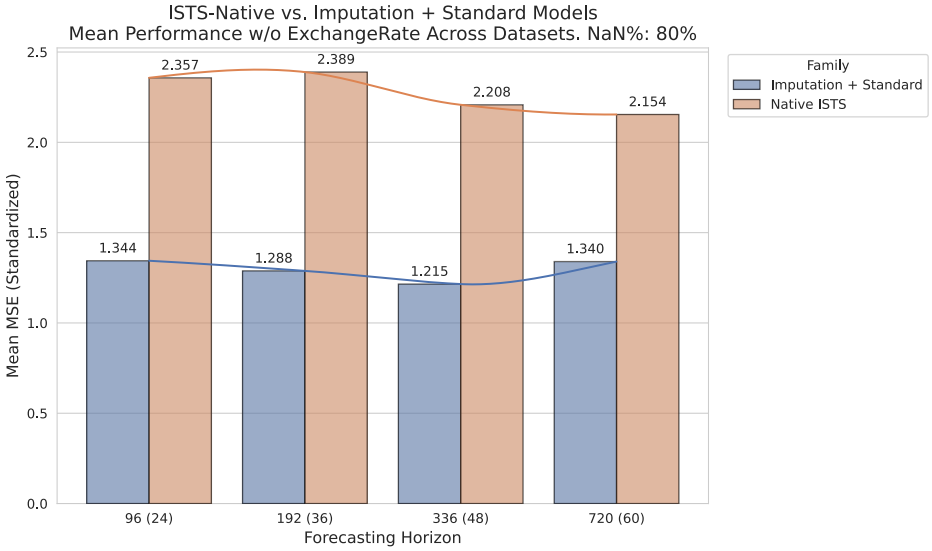


(c)

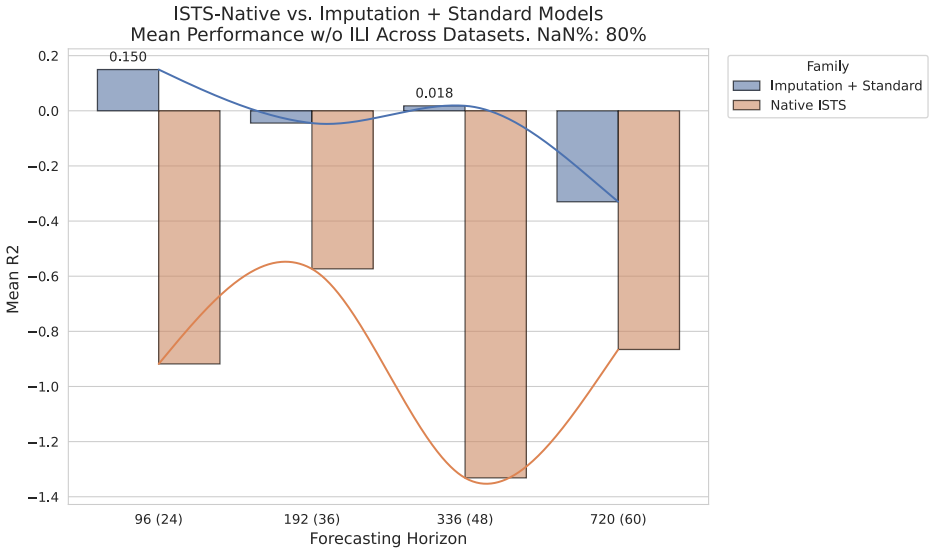
Figure A.30: Grouped barplots showing Native ISTS models performance vs Imputation + Standard model pipelines on all datasets **excluding** ExchangeRate with 50% missing data.



(a)

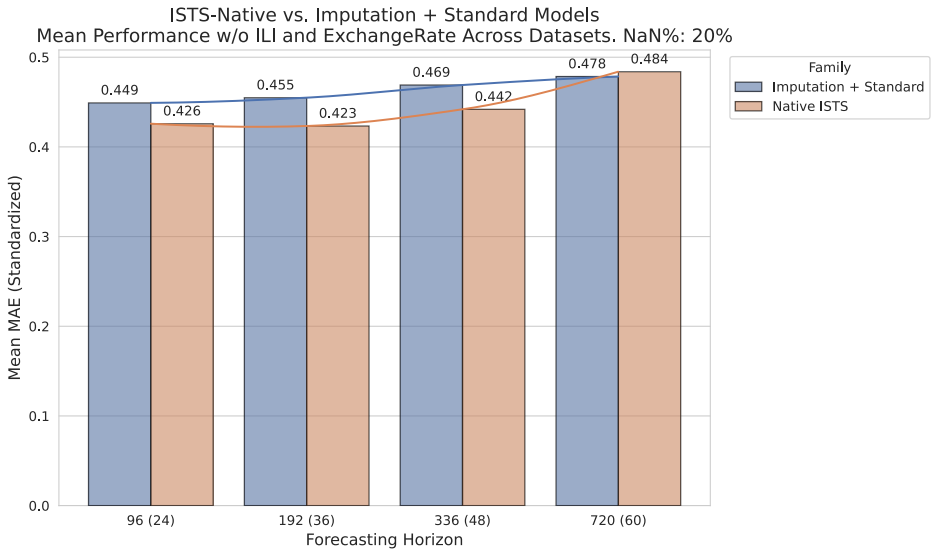


(b)

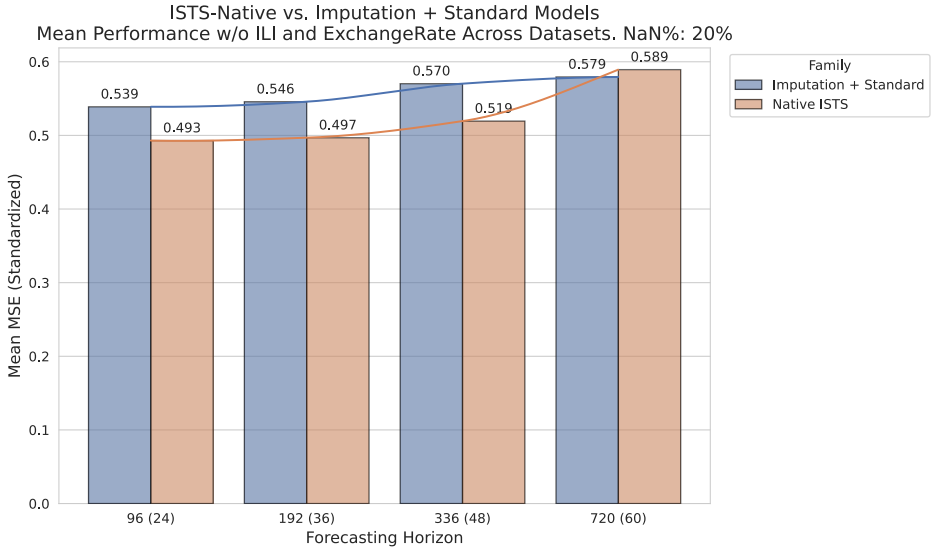


(c)

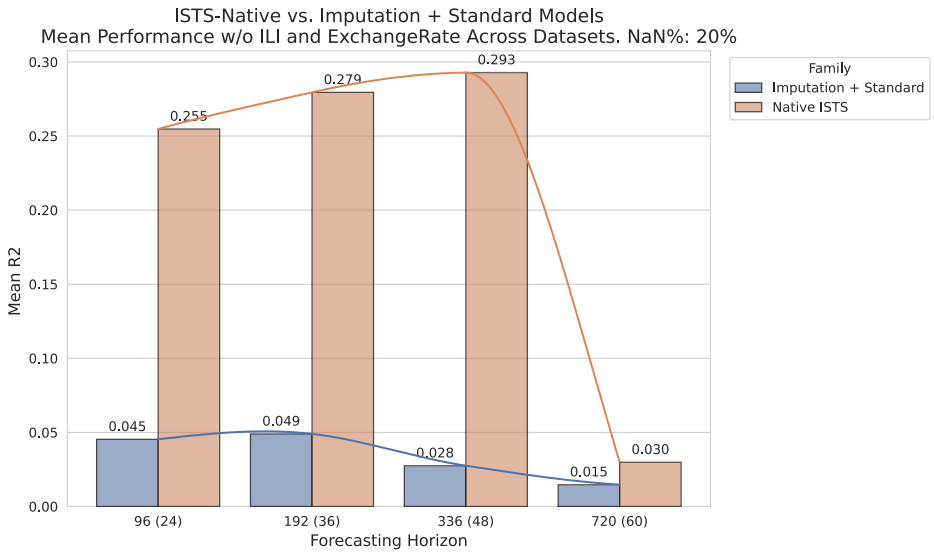
Figure A.31: Grouped barplots showing Native ISTS models performance vs Imputation + Standard model pipelines on all datasets **excluding** ExchangeRate with 80% missing data.



(a)

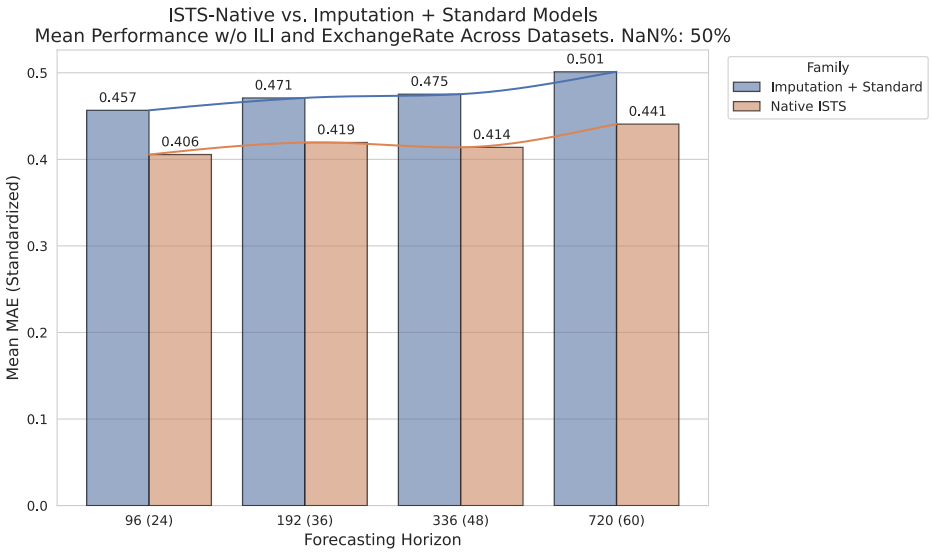


(b)

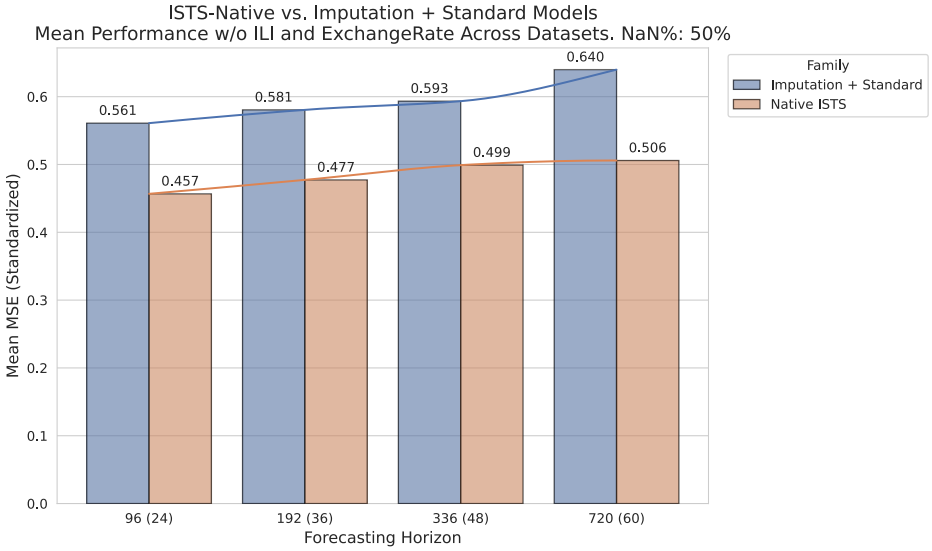


(c)

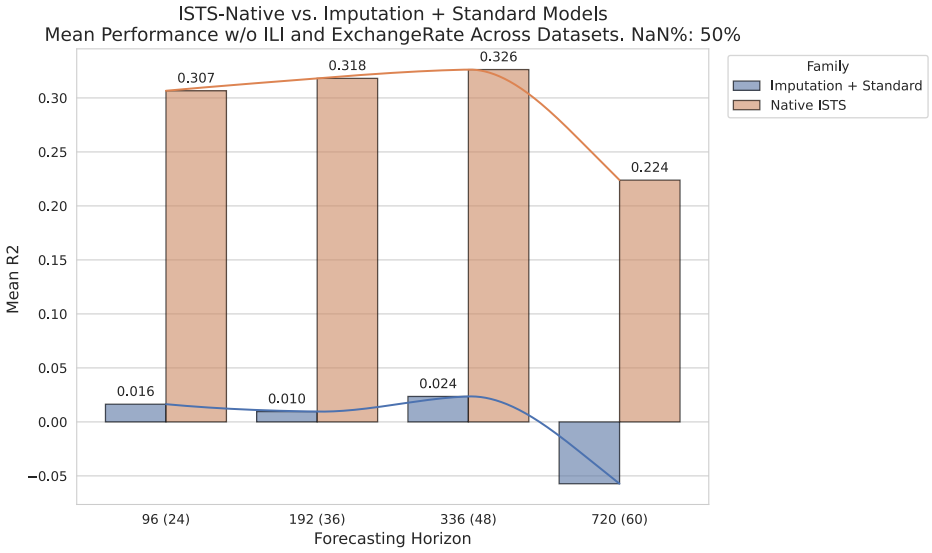
Figure A.32: Grouped barplots showing Native ISTS models performance vs Imputation + Standard model pipelines on all datasets **excluding** ILI and ExchangeRate with 20% missing data.



(a)

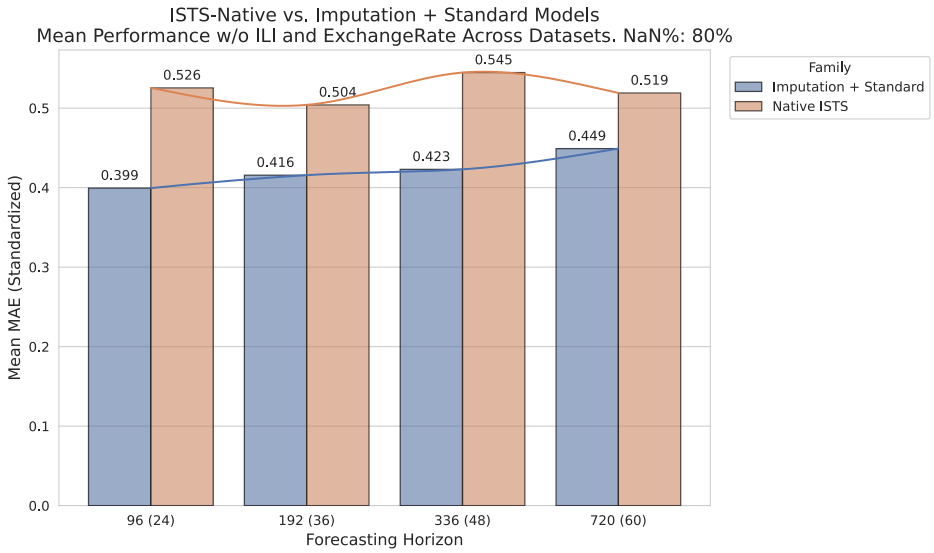


(b)

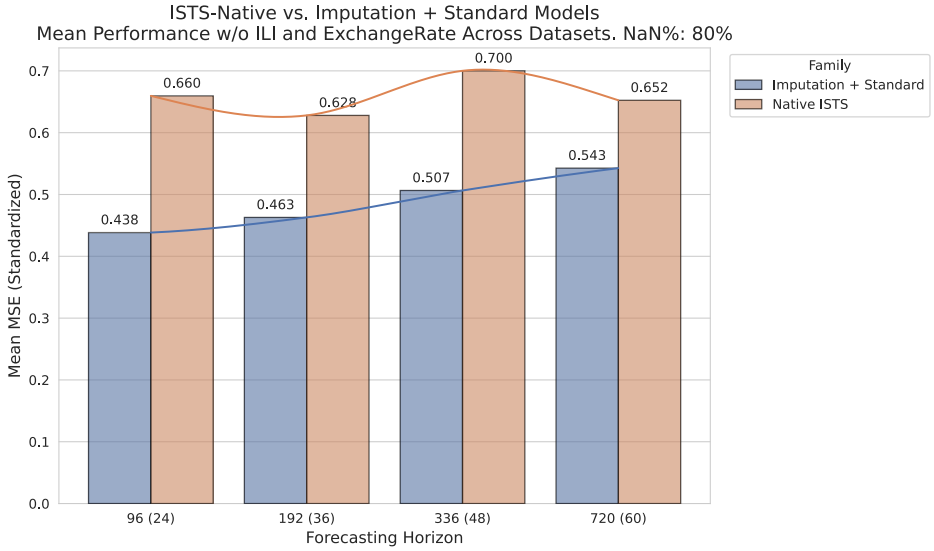


(c)

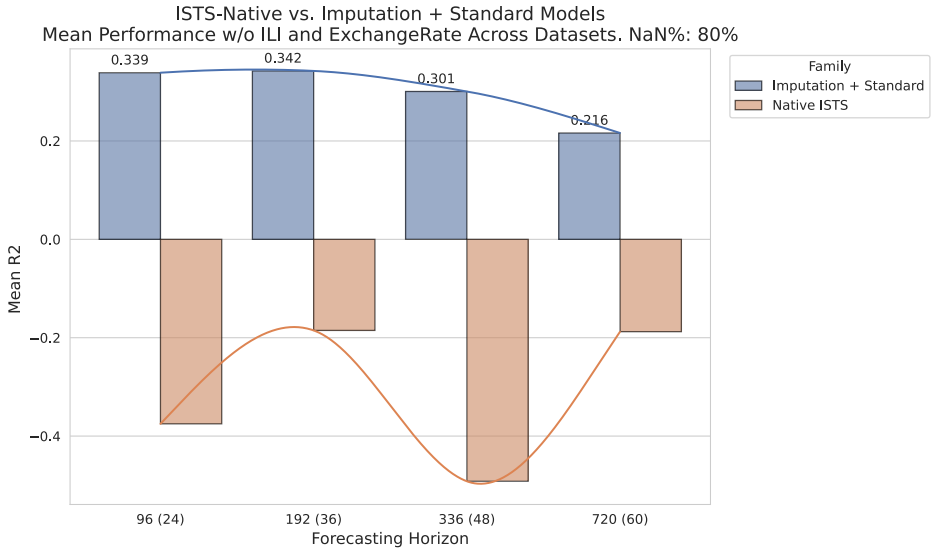
Figure A.33: Grouped barplots showing Native ISTS models performance vs Imputation + Standard model pipelines on all datasets **excluding** ILI and ExchangeRate with 50% missing data.



(a)



(b)



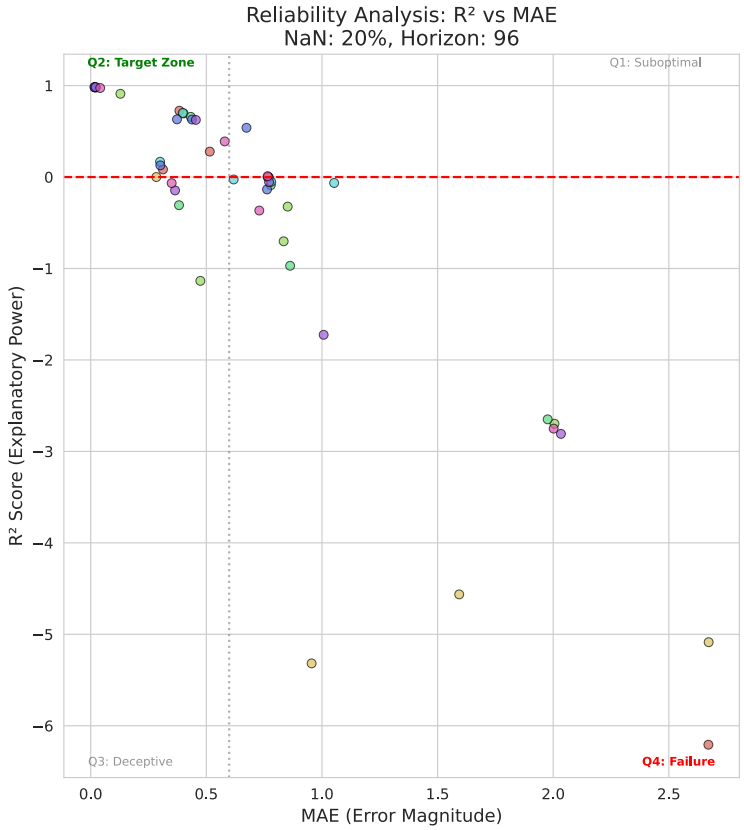
(c)

Figure A.34: Grouped barplots showing Native ISTS models performance vs Imputation + Standard model pipelines on all datasets **excluding** ILI and ExchangeRate with 50% missing data.

A.3.5 R2 Quadrants

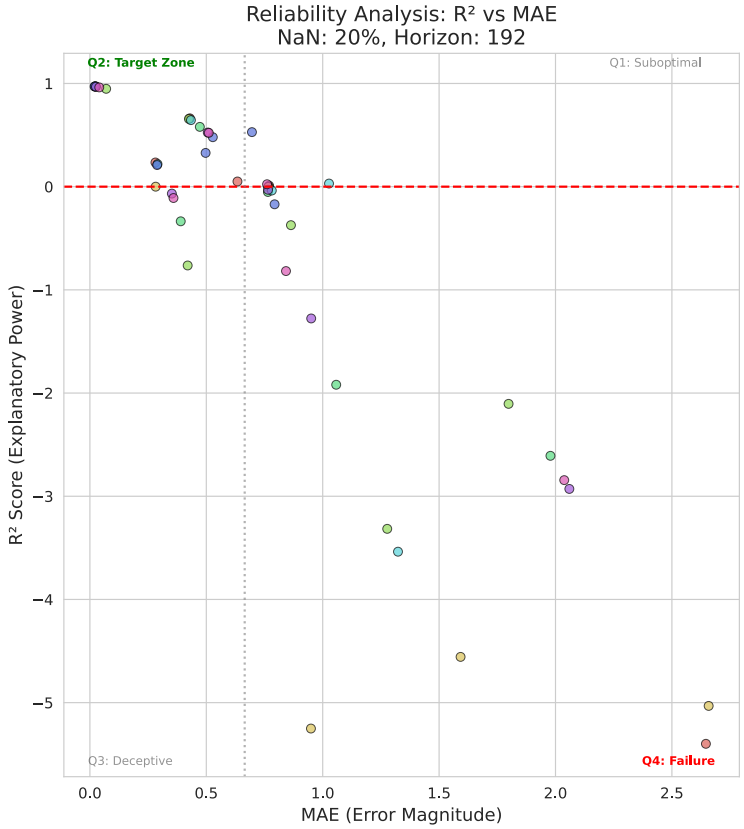
The following scatter plots categorize model performance into four behavioral quadrants based on their R^2 scores across various horizons and sparsity levels. This analysis is critical for distinguishing between models that achieve low error through "lucky" statistical averaging and those that successfully learn the underlying temporal signal.

The plots illustrate the relationship between error metrics and reliability, with a specific focus on identifying the "Failure Regime" (where $R^2 < 0$). These visuals highlight how increased data missingness (moving from 20% to 80% NaN) pushes models from the "Target Zone" (High R^2) toward the "Unreliable Zone" (Negative R^2), providing a visual confirmation of the predictive ceiling encountered in sparse datasets.



Quadrant	Count
Q1: Suboptimal	2
Q2: Target	19
Q3: Deceptive	5
Q4: Failure	22

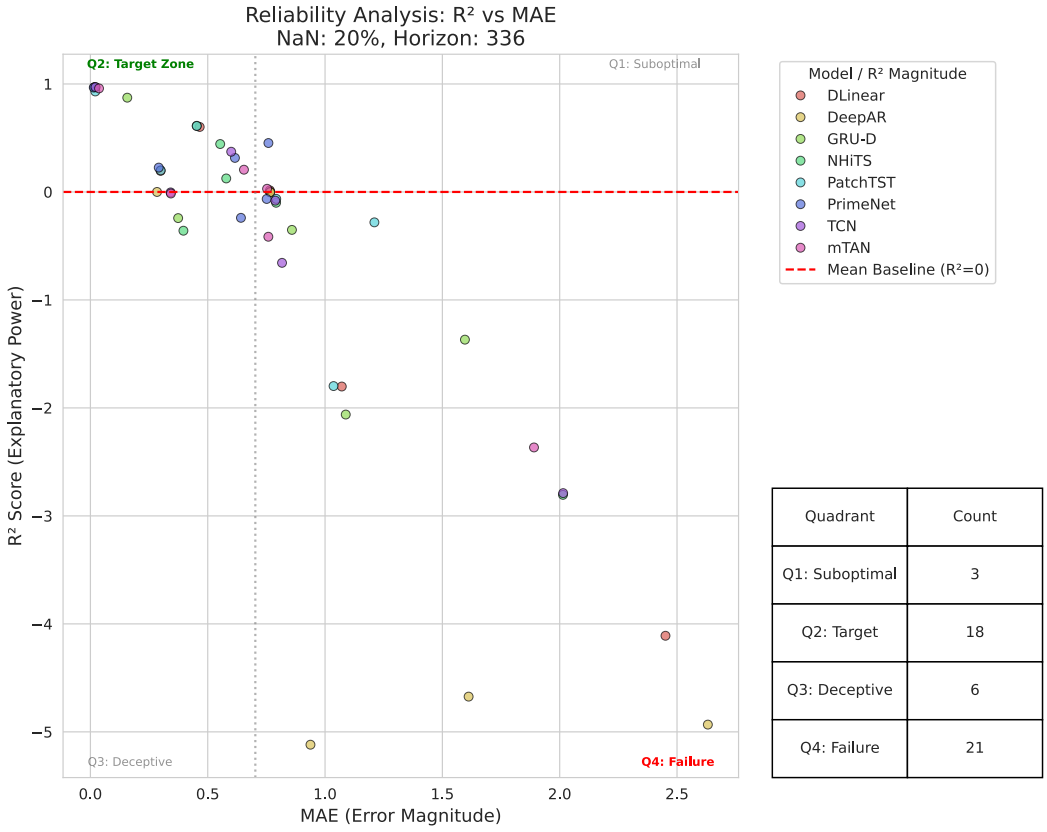
(a)



Quadrant	Count
Q1: Suboptimal	4
Q2: Target	19
Q3: Deceptive	5
Q4: Failure	20

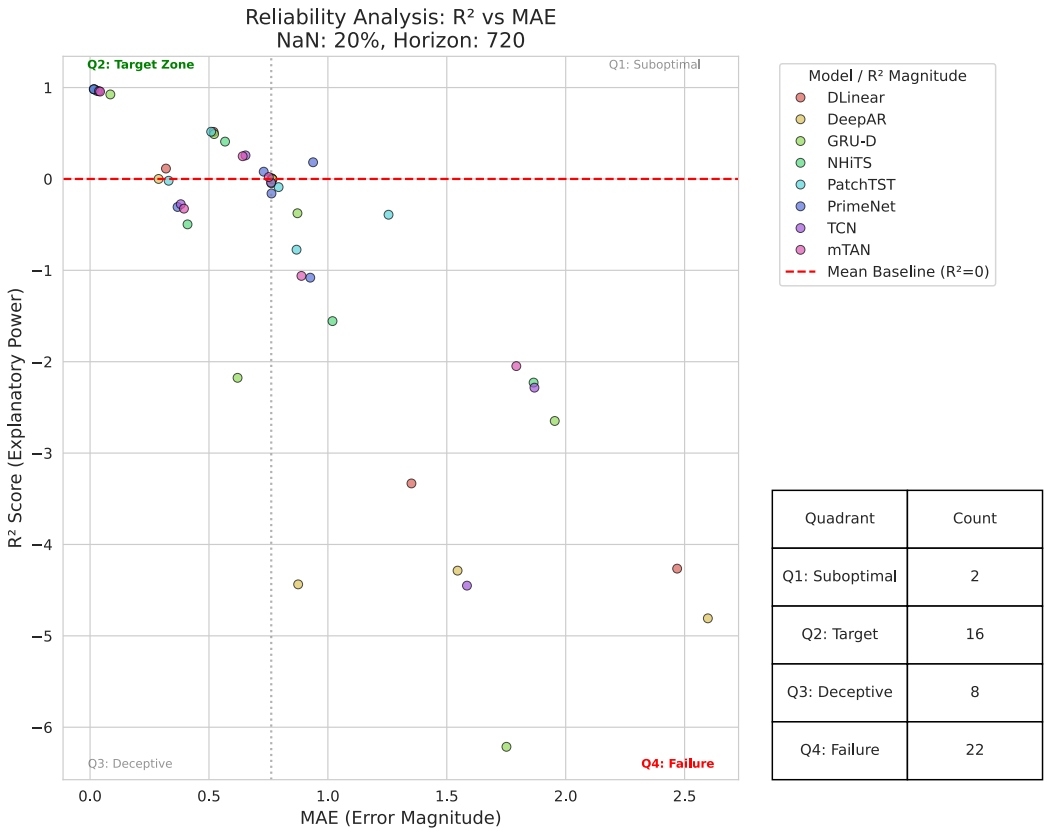
(b)

Figure A.35: R^2 Score scatter comparing the trustworthyness of models' prediction with 20% missing data, part 1.



Quadrant	Count
Q1: Suboptimal	3
Q2: Target	18
Q3: Deceptive	6
Q4: Failure	21

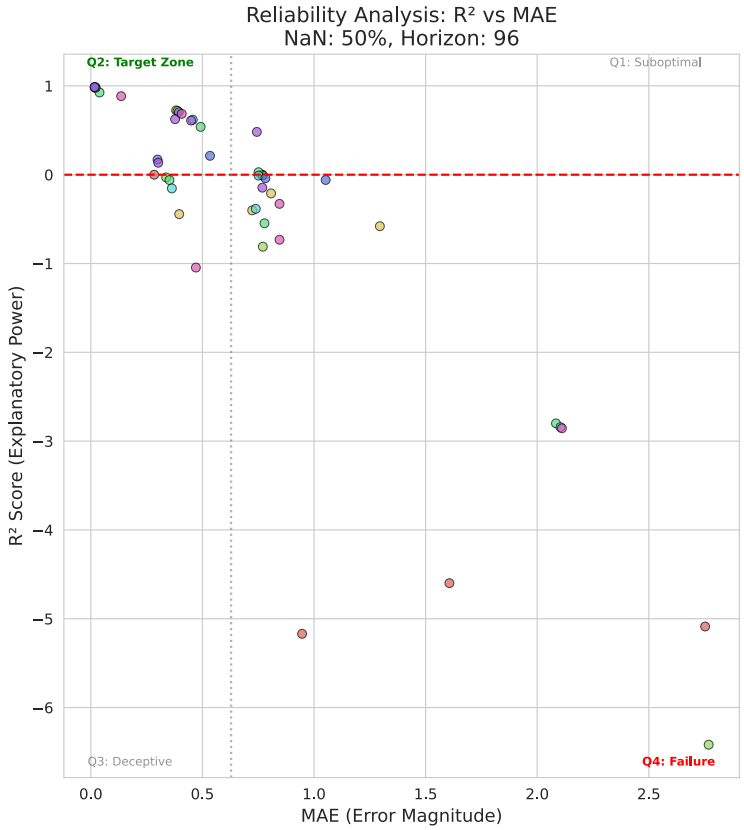
(a)



Quadrant	Count
Q1: Suboptimal	2
Q2: Target	16
Q3: Deceptive	8
Q4: Failure	22

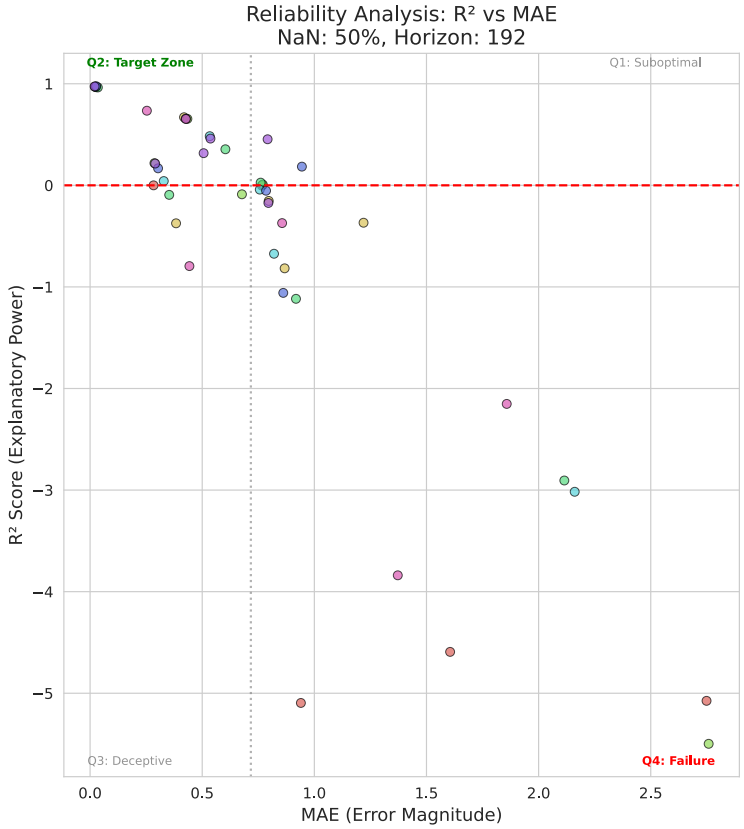
(b)

Figure A.36: R^2 Score scatter comparing the trustworthyness of models' prediction with 20% missing data, part 2.



Quadrant	Count
Q1: Suboptimal	2
Q2: Target	18
Q3: Deceptive	6
Q4: Failure	22

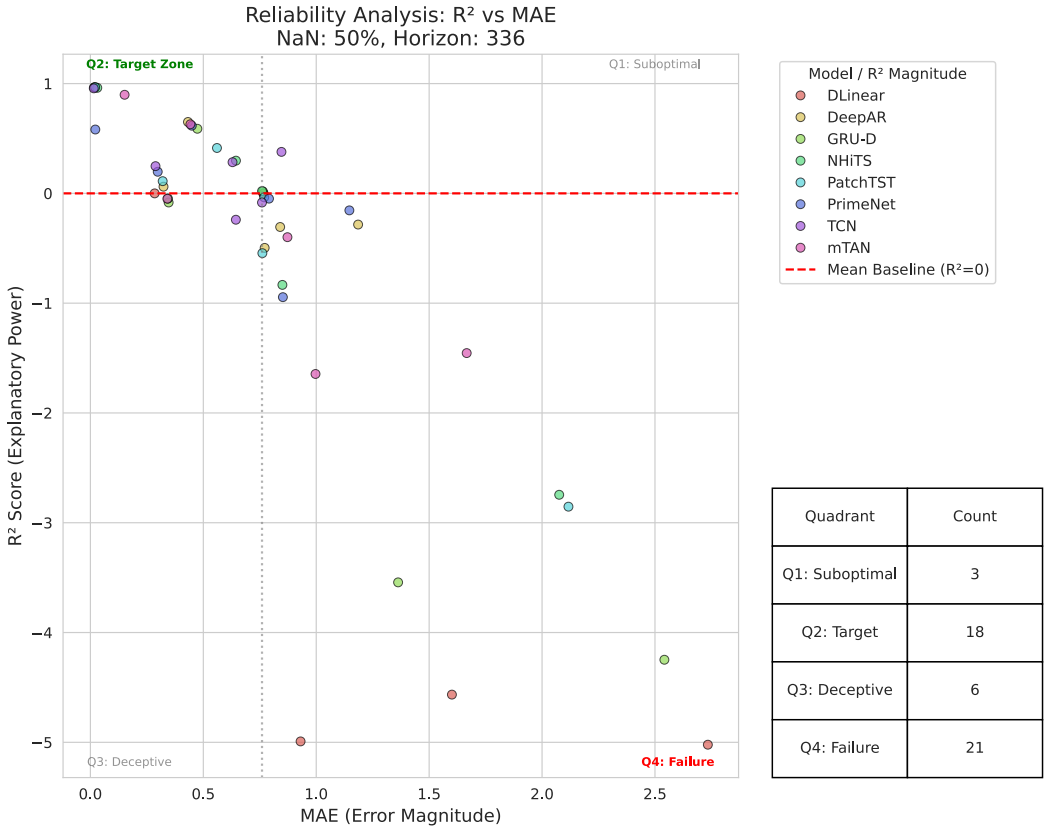
(a)



Quadrant	Count
Q1: Suboptimal	4
Q2: Target	19
Q3: Deceptive	5
Q4: Failure	20

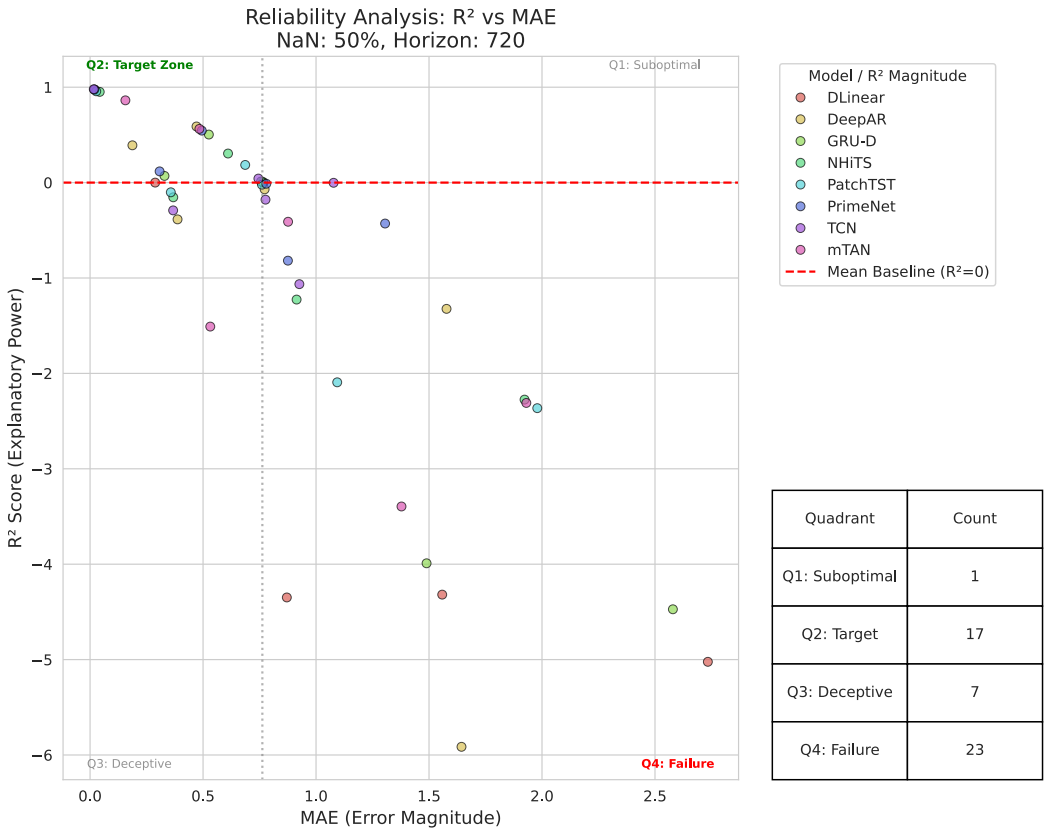
(b)

Figure A.37: R^2 Score scatter comparing the trustworthyness of models' prediction with 50% missing data, part 1.



Quadrant	Count
Q1: Suboptimal	3
Q2: Target	18
Q3: Deceptive	6
Q4: Failure	21

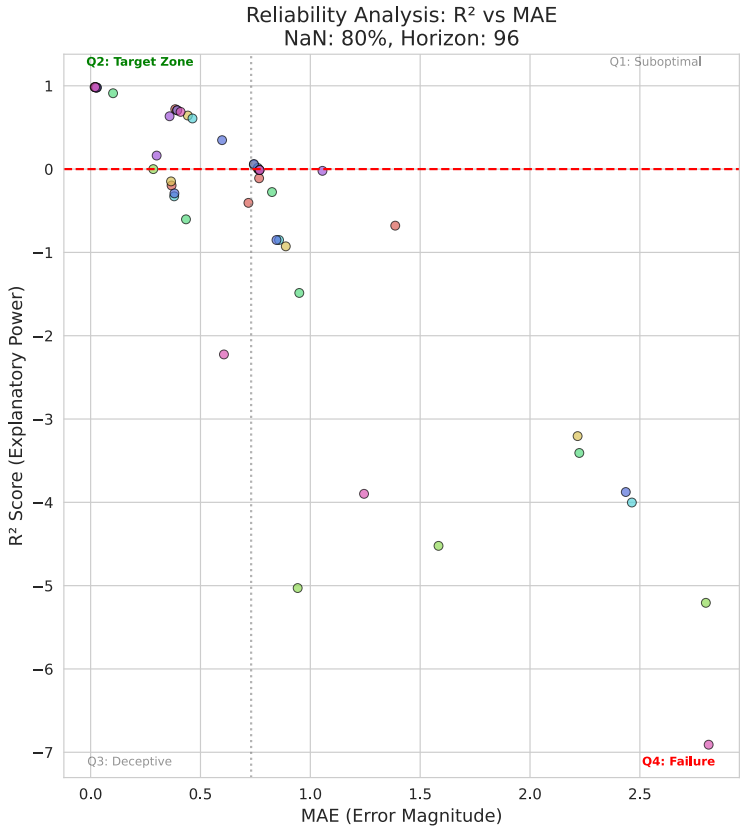
(a)



Quadrant	Count
Q1: Suboptimal	1
Q2: Target	17
Q3: Deceptive	7
Q4: Failure	23

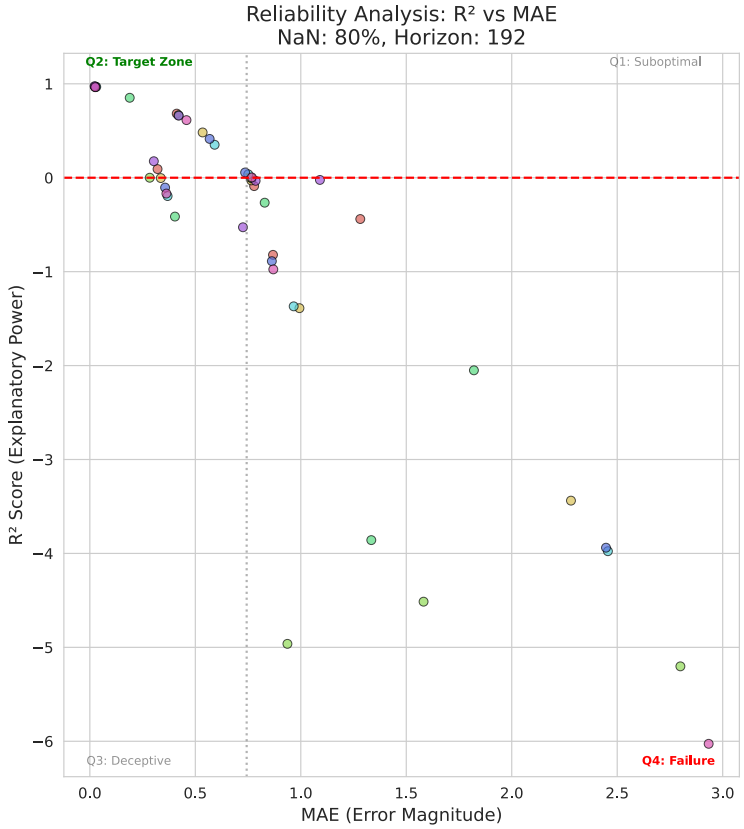
(b)

Figure A.38: R^2 Score scatter comparing the trustworthyness of models' prediction with 50% missing data, part 2.



Quadrant	Count
Q1: Suboptimal	3
Q2: Target	16
Q3: Deceptive	8
Q4: Failure	21

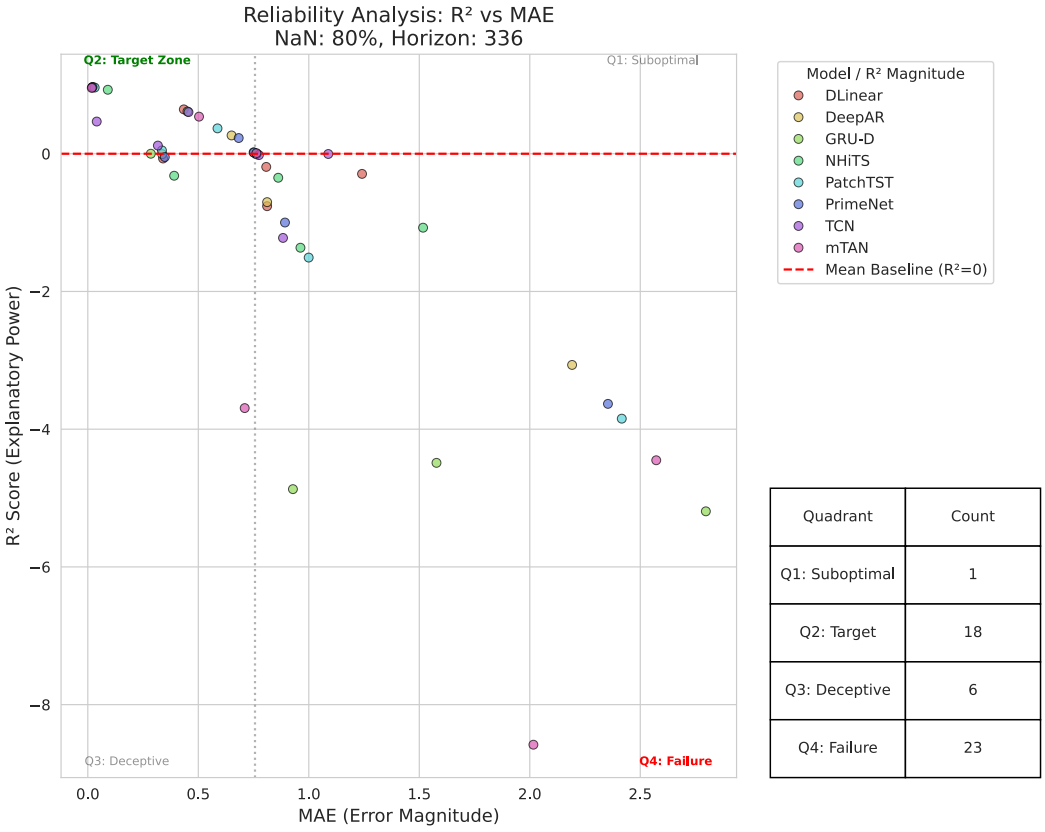
(a)



Quadrant	Count
Q1: Suboptimal	2
Q2: Target	17
Q3: Deceptive	7
Q4: Failure	22

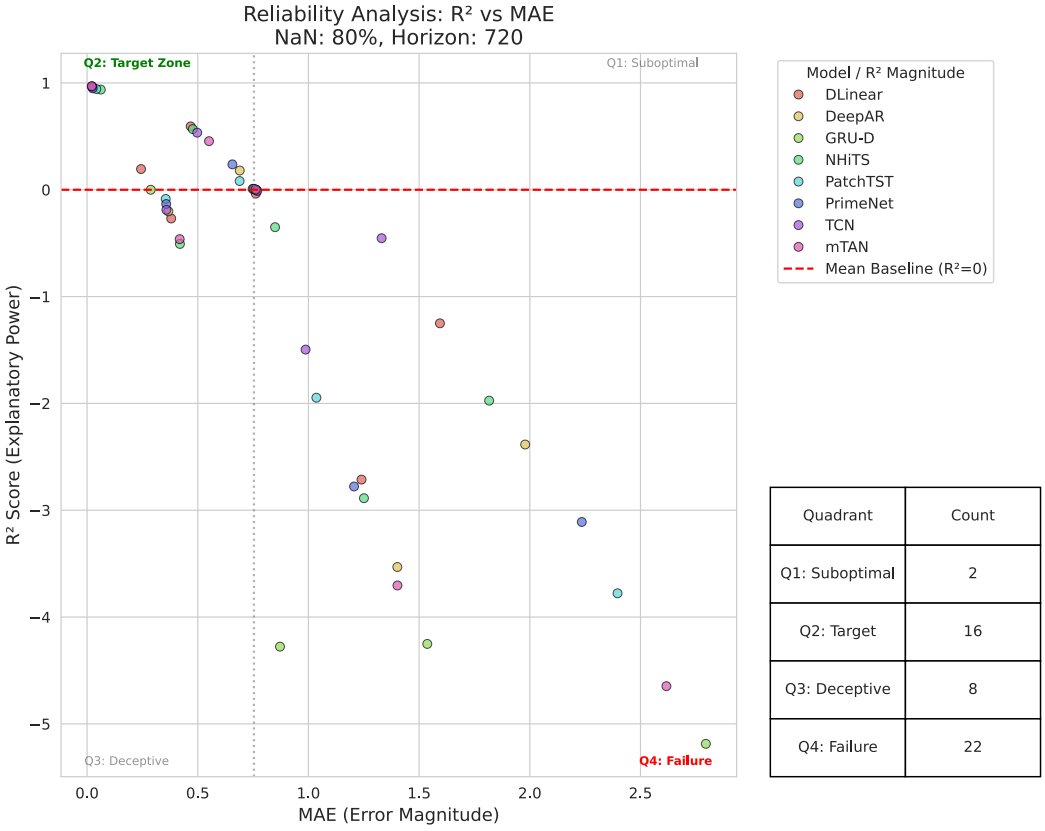
(b)

Figure A.39: R^2 Score scatter comparing the trustworthyness of models' prediction with 80% missing data, part 1.



Quadrant	Count
Q1: Suboptimal	1
Q2: Target	18
Q3: Deceptive	6
Q4: Failure	23

(a)



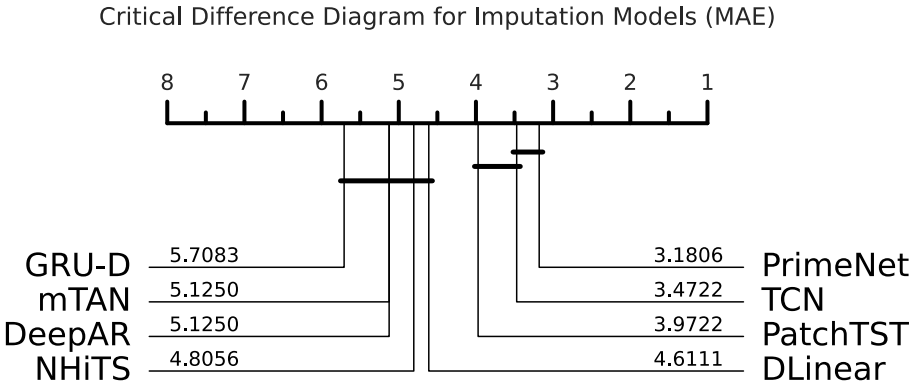
Quadrant	Count
Q1: Suboptimal	2
Q2: Target	16
Q3: Deceptive	8
Q4: Failure	22

(b)

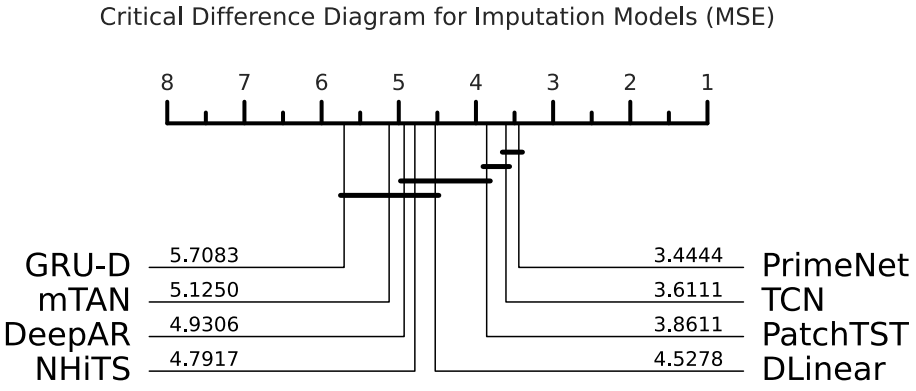
Figure A.40: R^2 Score scatter comparing the trustworthyness of models' prediction with 80% missing data, part 2.

A.3.6 Critical Diagrams

CD Diagrams visualize the results of the Nemenyi post-hoc test. Models connected by a horizontal bar are not statistically different at $\alpha = 0.05$.

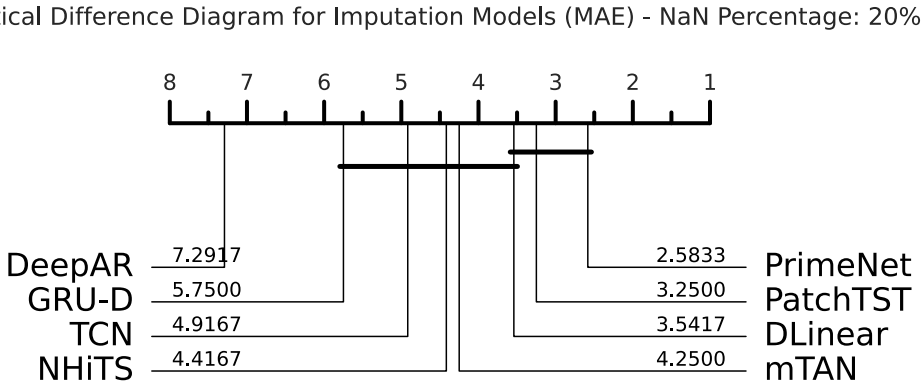


(a)

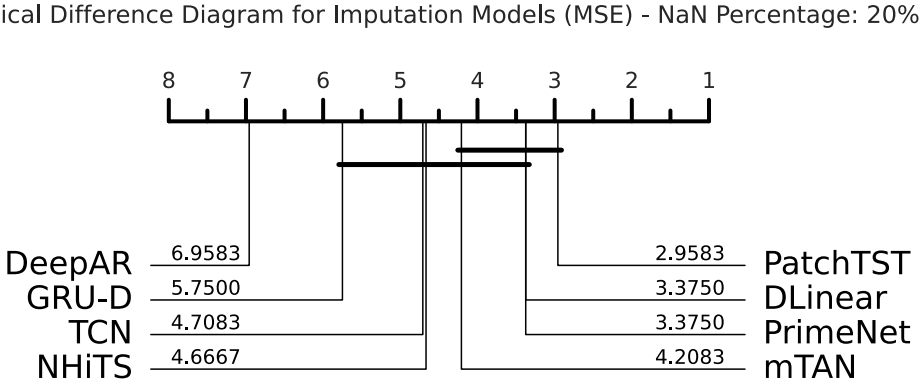


(b)

Figure A.41: Critical Difference diagrams on MAE and MSE across all NaN percentage scenarios.

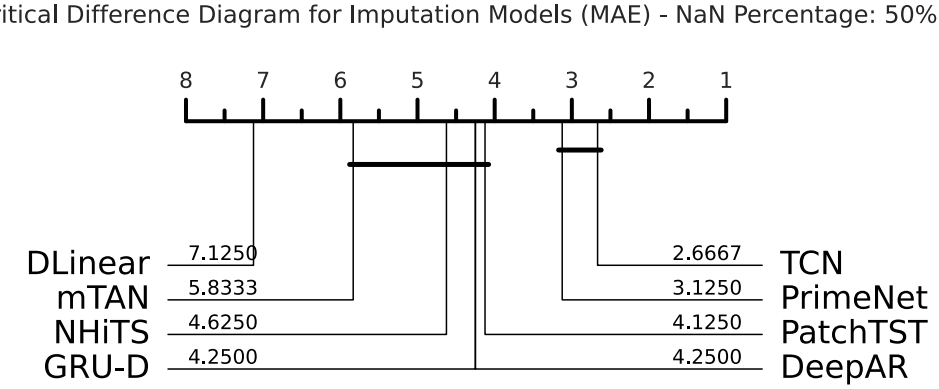


(a)

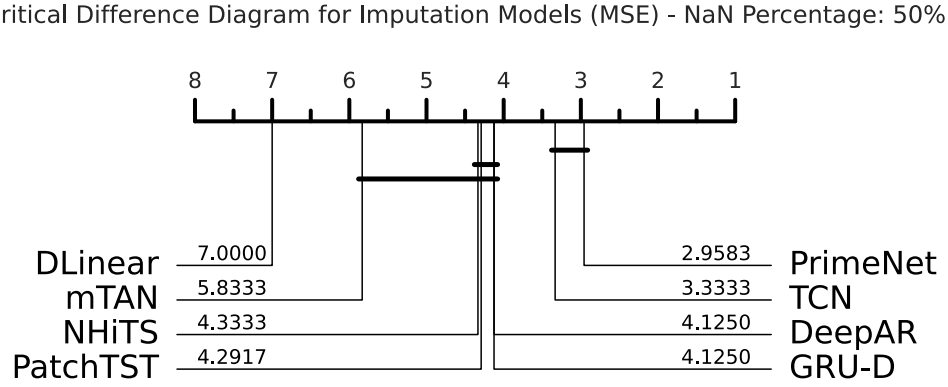


(b)

Figure A.42: Critical Difference diagrams on MAE and MSE with 20% missing data.

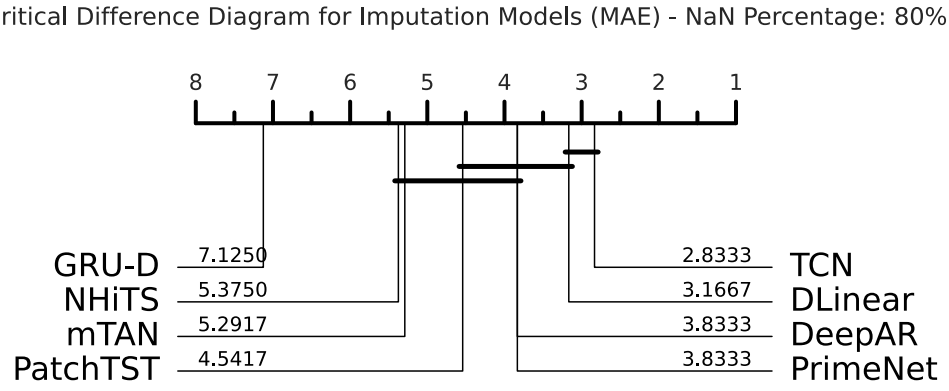


(a)

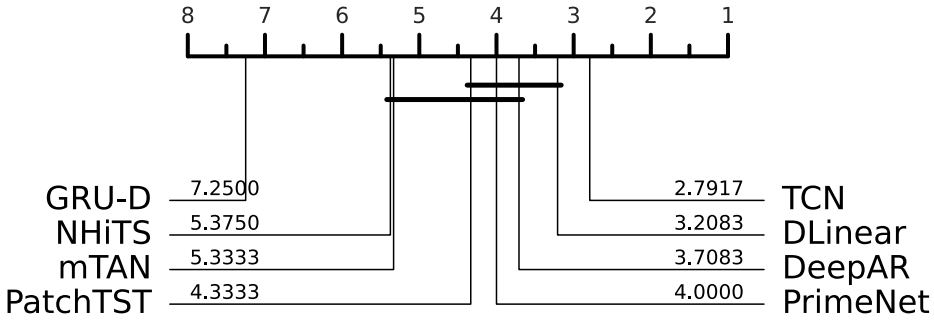


(b)

Figure A.43: Critical Difference diagrams on MAE and MSE with 50% missing data.



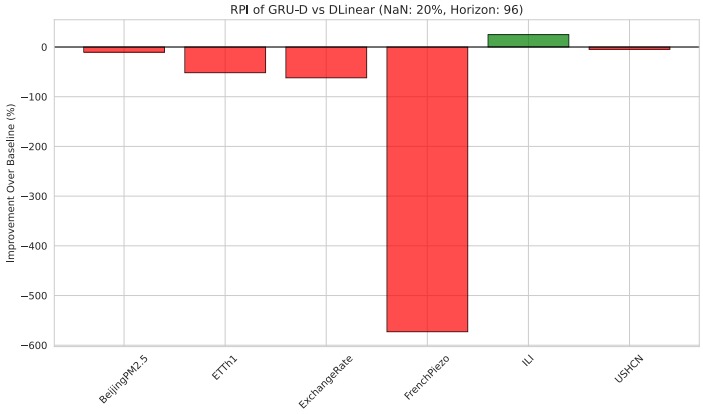
(a)



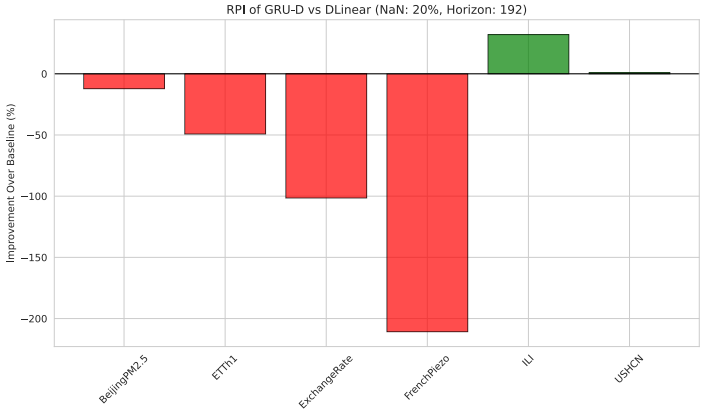
(b)

Figure A.44: Critical Difference diagrams on MAE and MSE with 80% missing data.

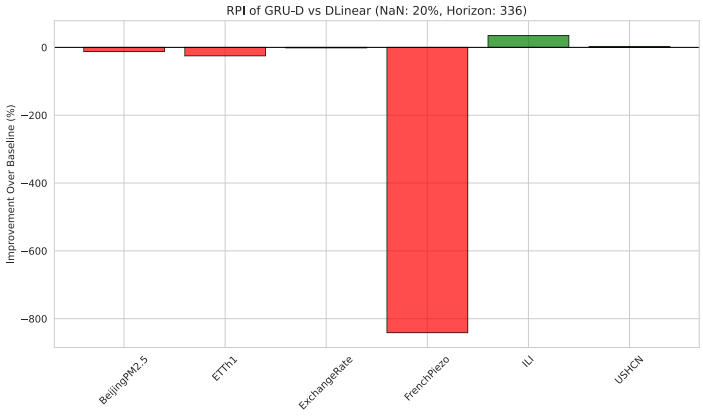
A.3.7 Relative Performance Improvement (RPI) Plots



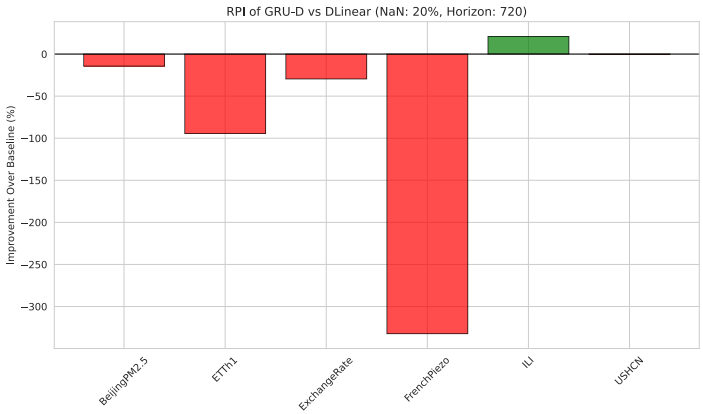
(a)



(b)

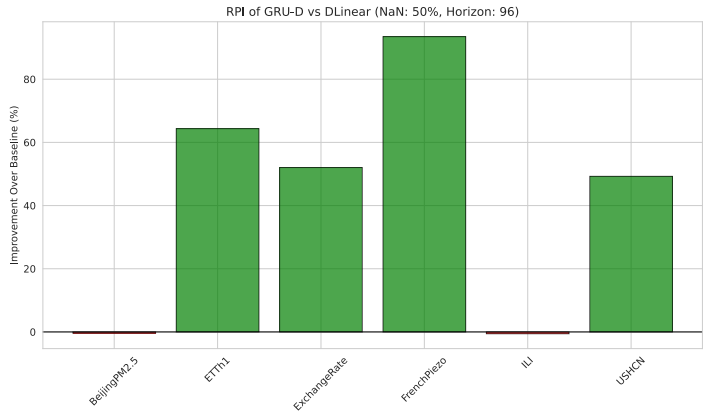


(c)

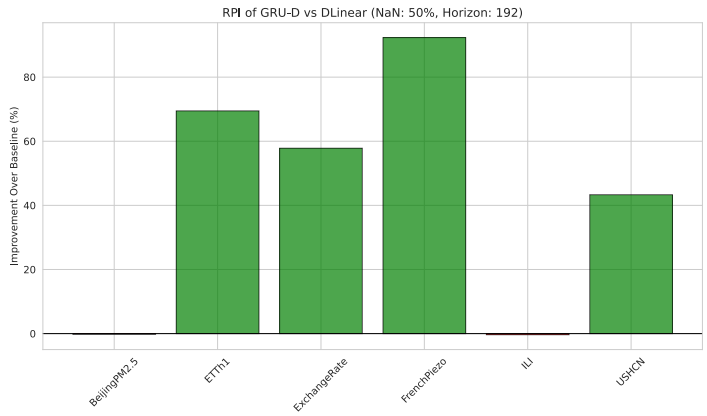


(d)

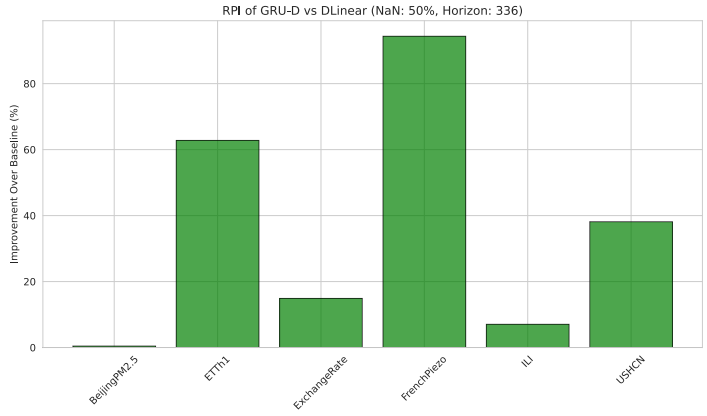
Figure A.45: Relative Performance Improvement of GRU-D over DLinear with 20% missing data.



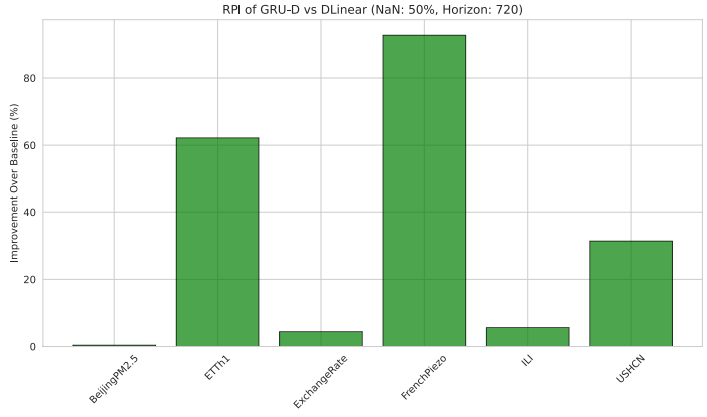
(a)



(b)

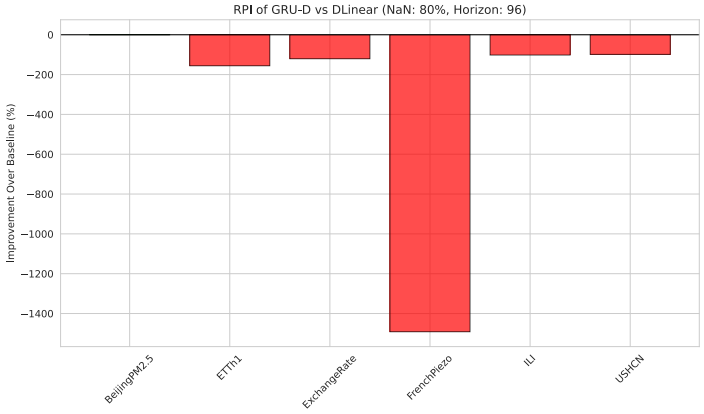


(c)

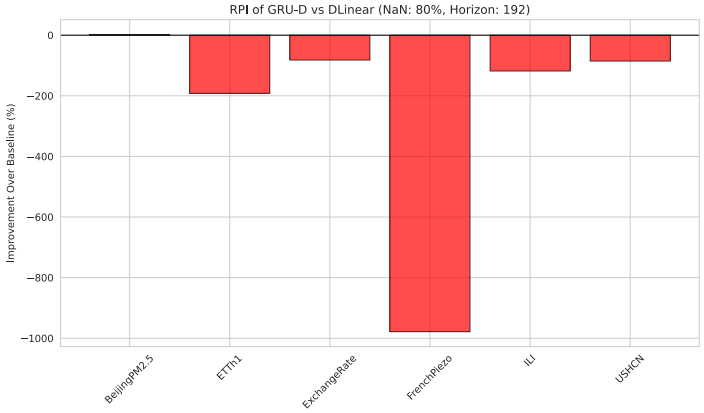


(d)

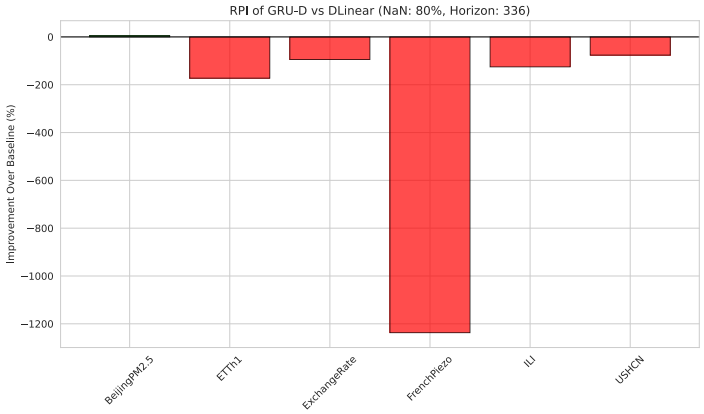
Figure A.46: Relative Performance Improvement of GRU-D over DLinear with 50% missing data.



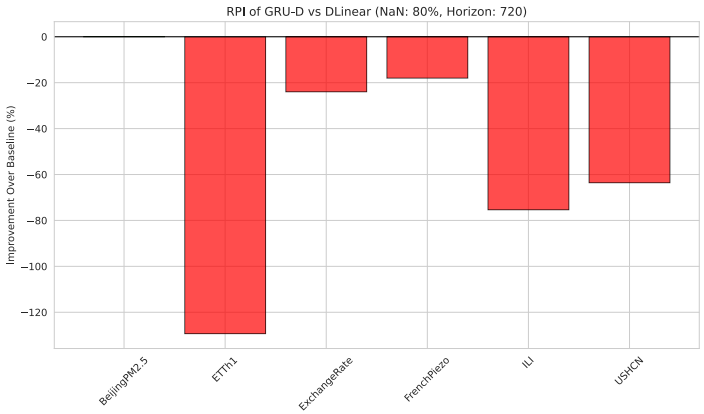
(a)



(b)

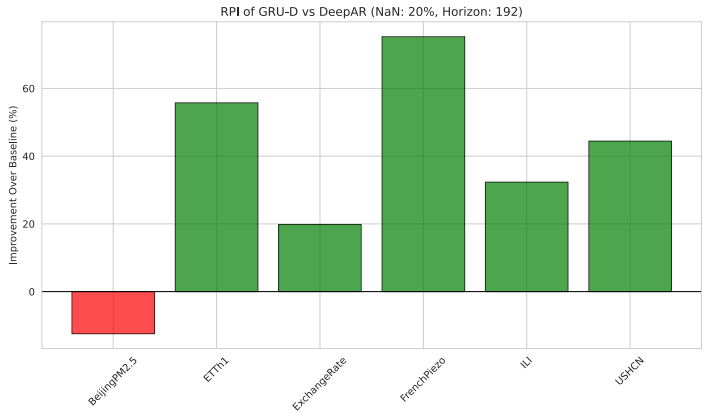


(c)

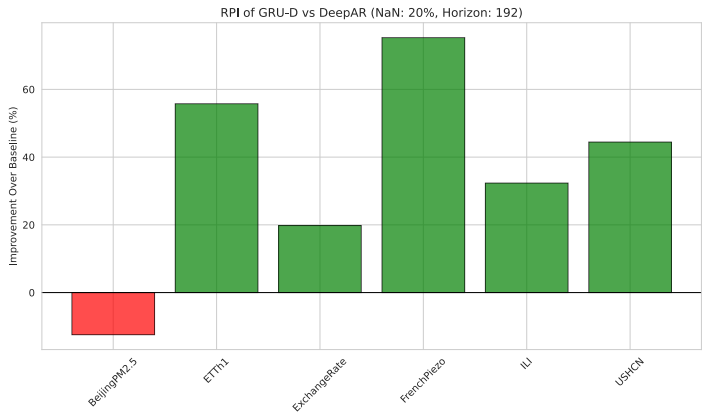


(d)

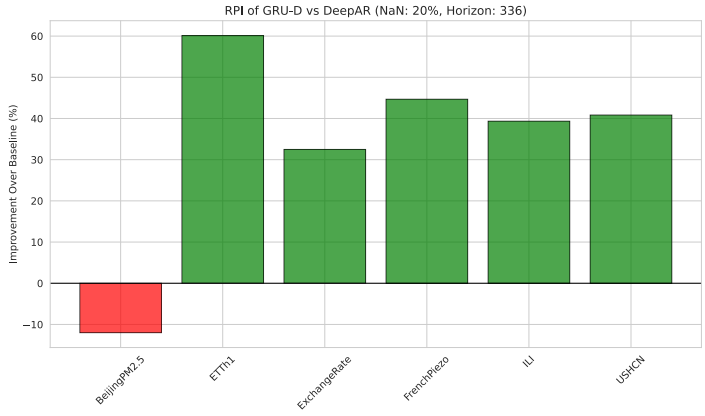
Figure A.47: Relative Performance Improvement of GRU-D over DLinear with 80% missing data.



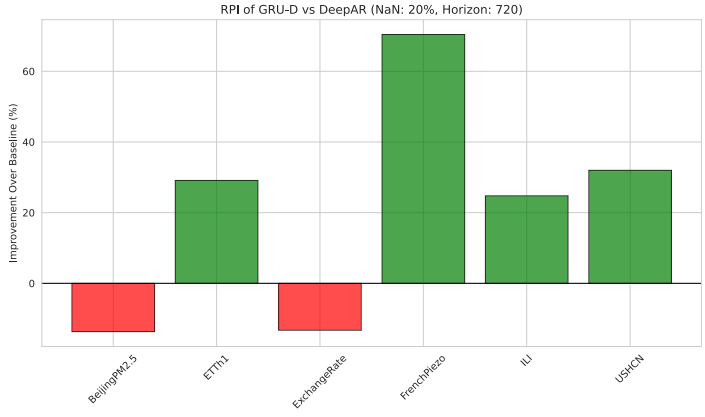
(a)



(b)

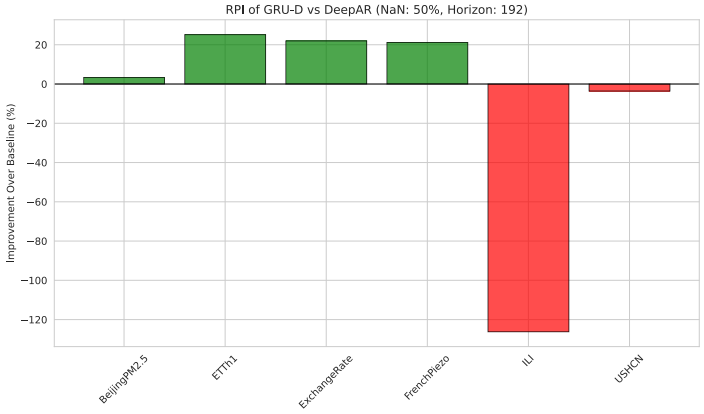


(c)

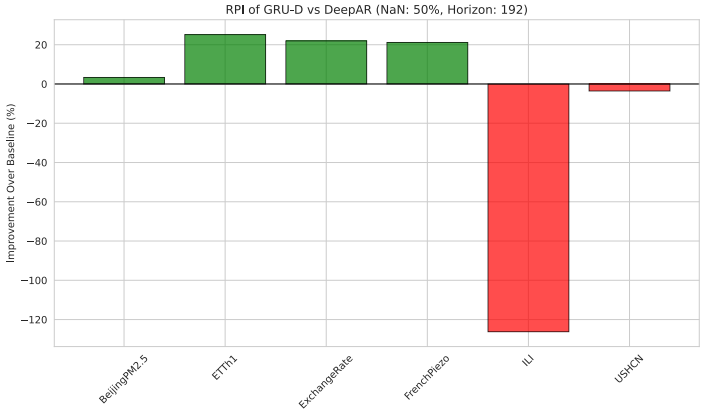


(d)

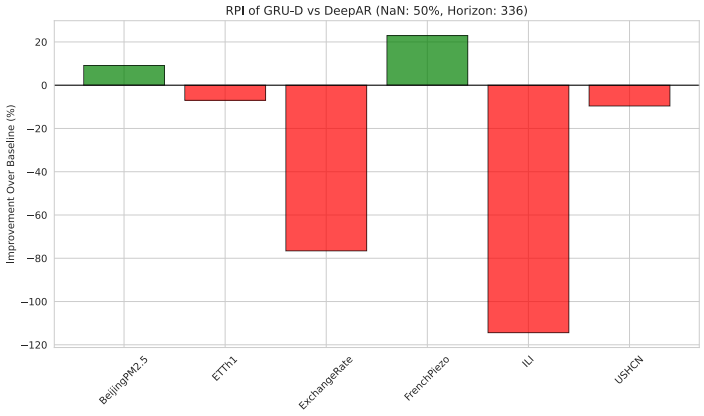
Figure A.48: Relative Performance Improvement of GRU-D over DeepAR with 20% missing data.



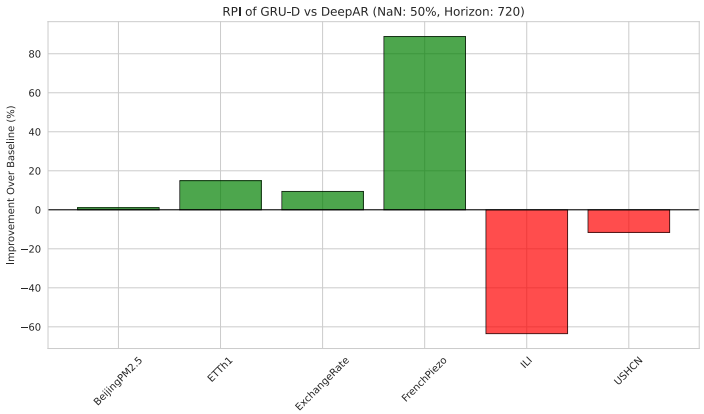
(a)



(b)

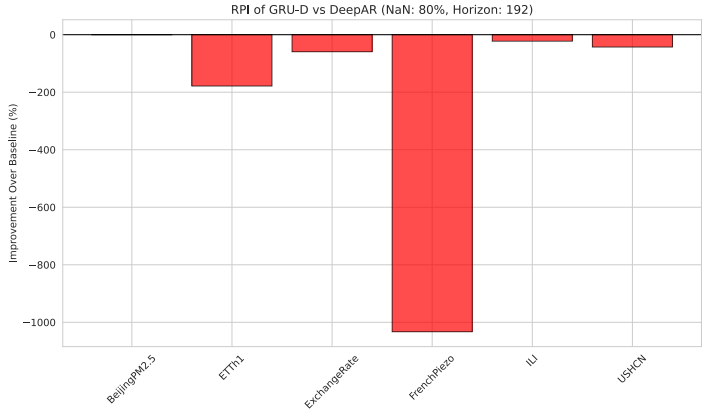


(c)

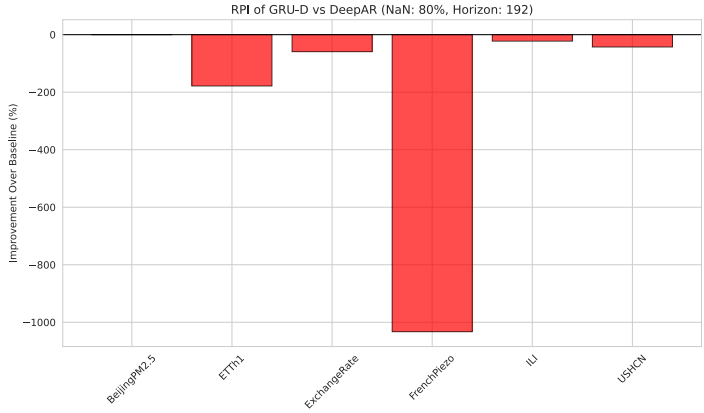


(d)

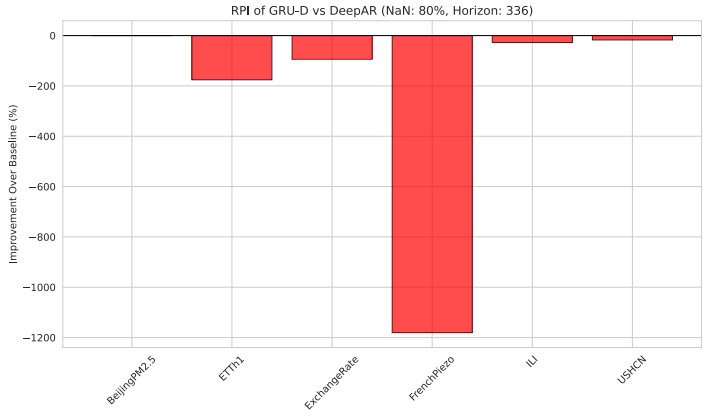
Figure A.49: Relative Performance Improvement of GRU-D over DeepAR with 50% missing data.



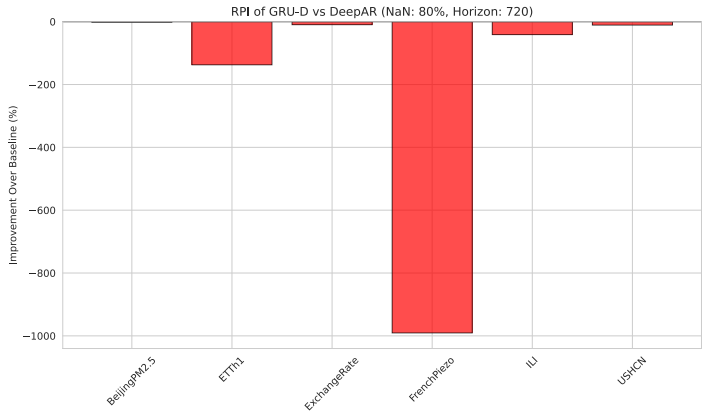
(a)



(b)

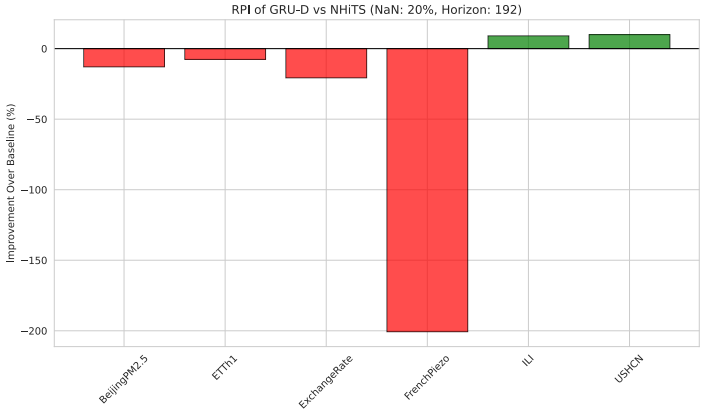


(c)

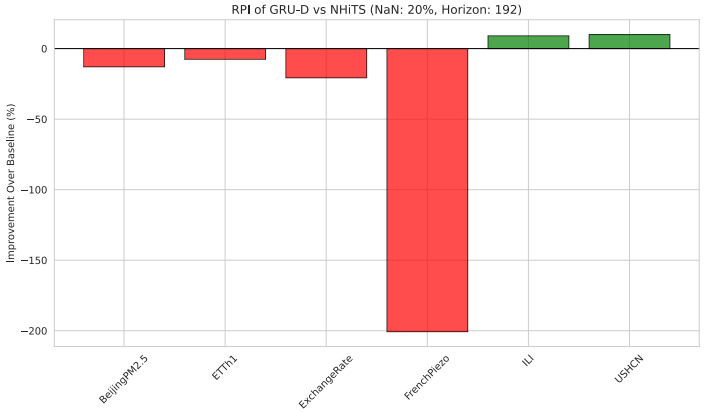


(d)

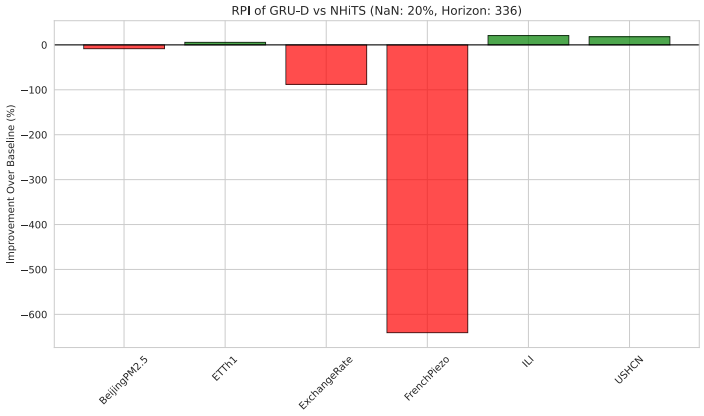
Figure A.50: Relative Performance Improvement of GRU-D over DeepAR with 80% missing data.



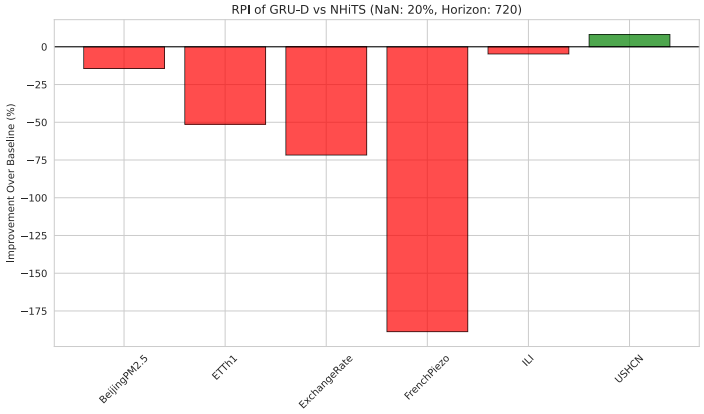
(a)



(b)

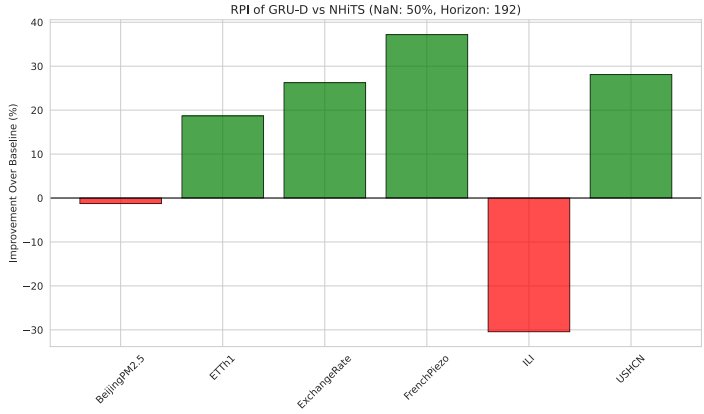


(c)

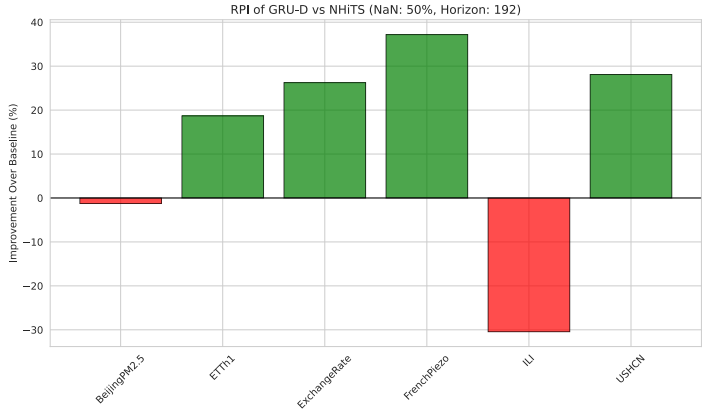


(d)

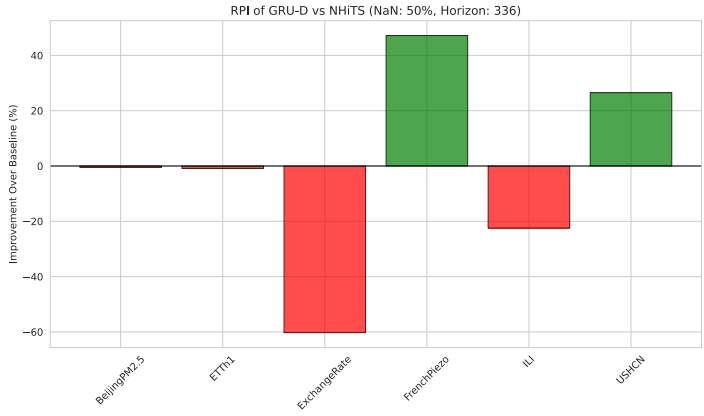
Figure A.51: Relative Performance Improvement of GRU-D over NHiTS with 20% missing data.



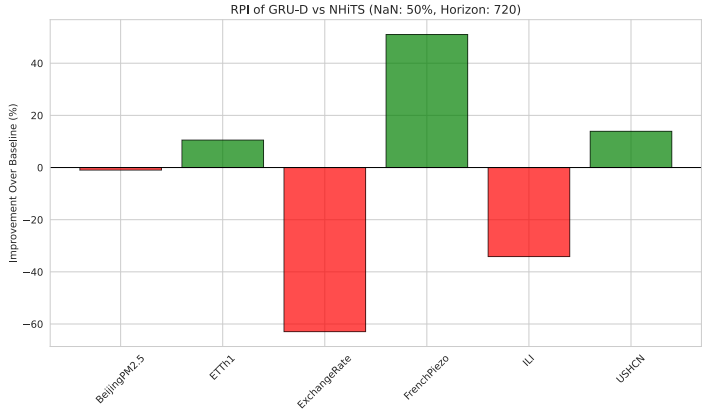
(a)



(b)

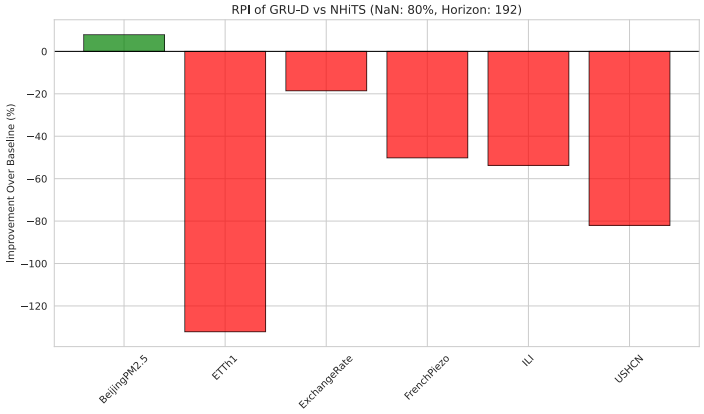


(c)

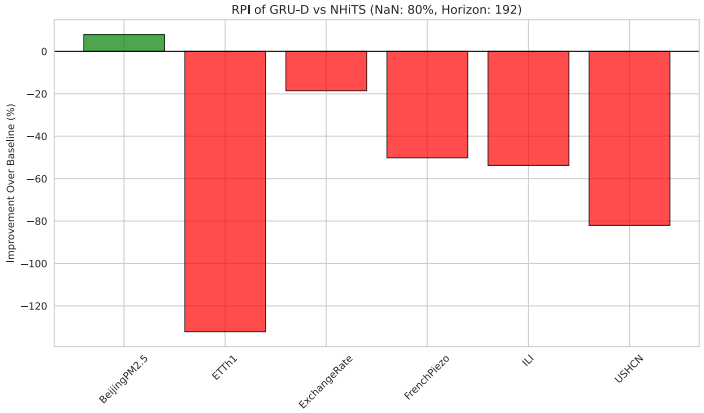


(d)

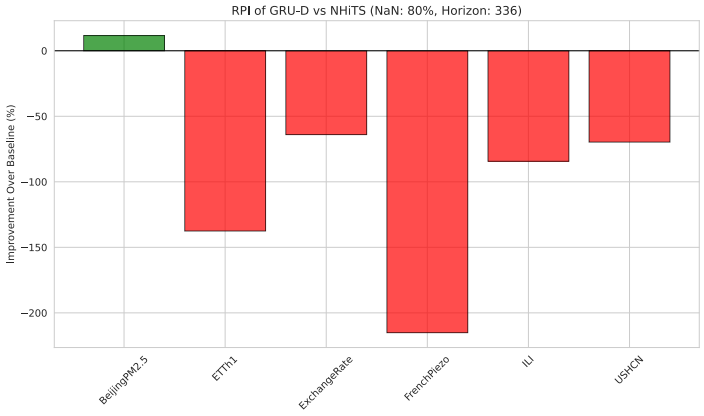
Figure A.52: Relative Performance Improvement of GRU-D over NHiTS with 50% missing data.



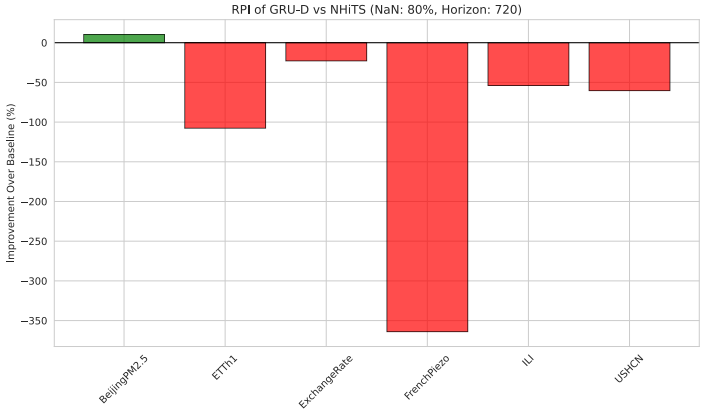
(a)



(b)

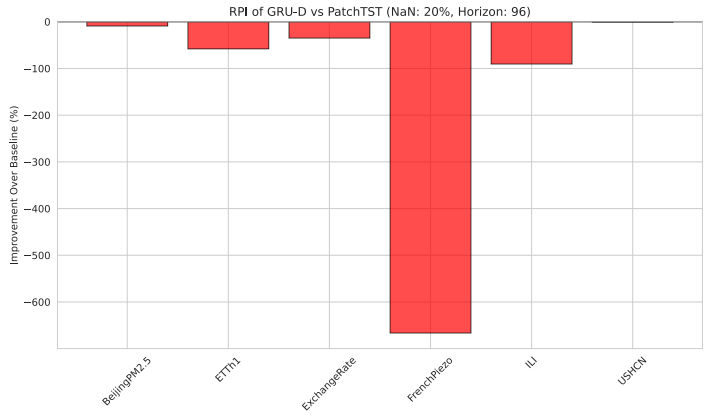


(c)

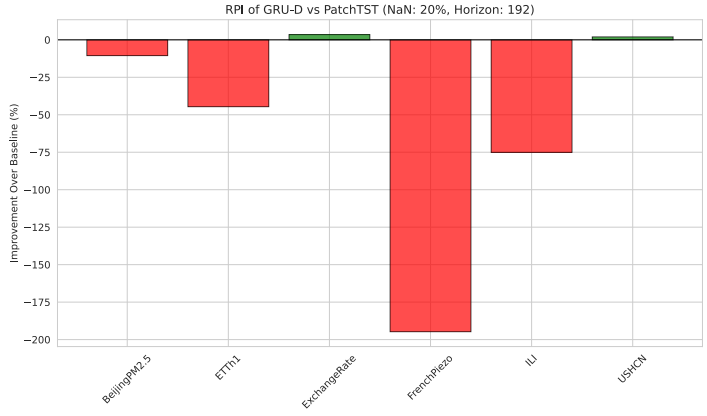


(d)

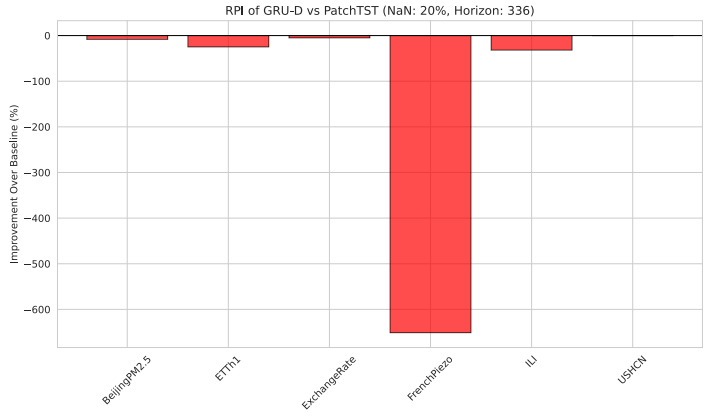
Figure A.53: Relative Performance Improvement of GRU-D over NHiTS with 80% missing data.



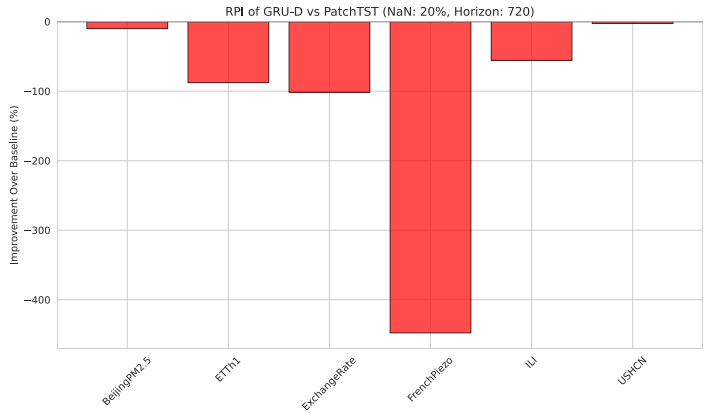
(a)



(b)

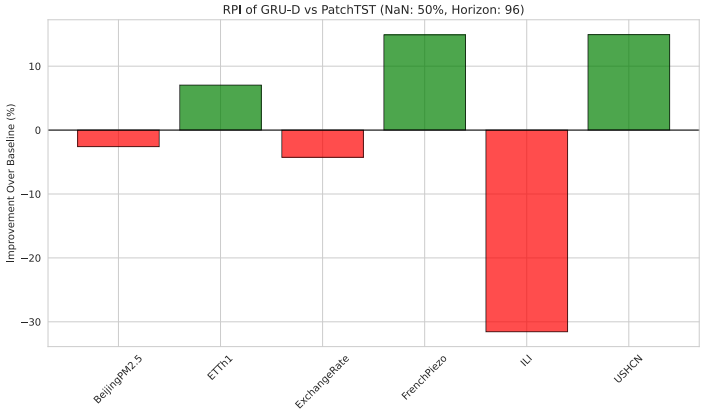


(c)

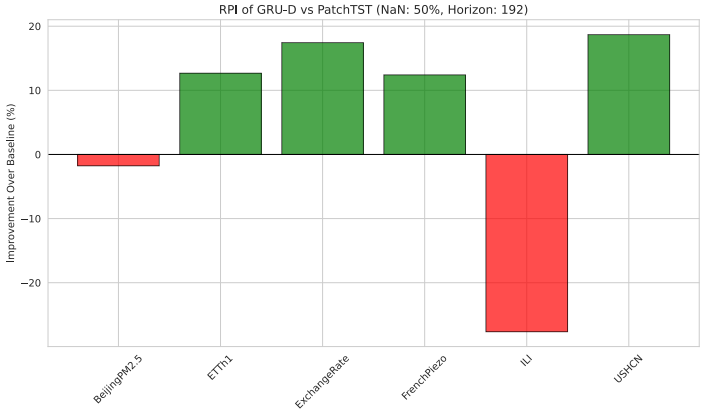


(d)

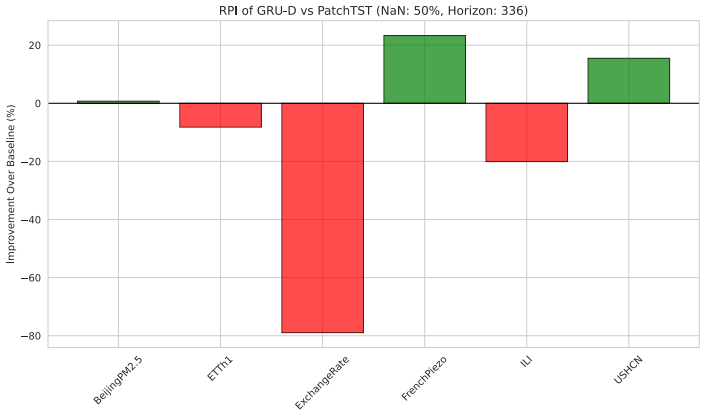
Figure A.54: Relative Performance Improvement of GRU-D over PatchTST with 20% missing data.



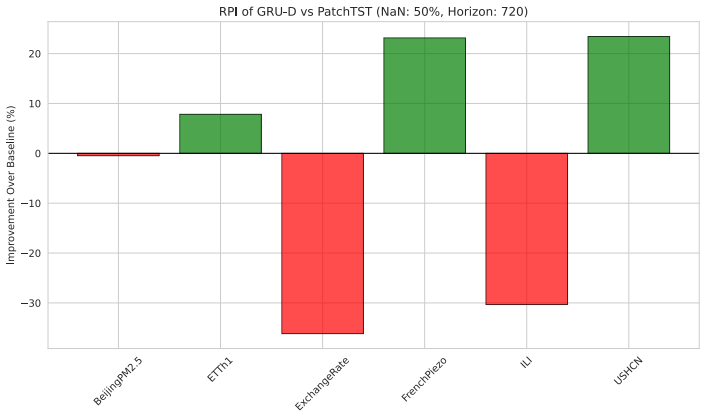
(a)



(b)

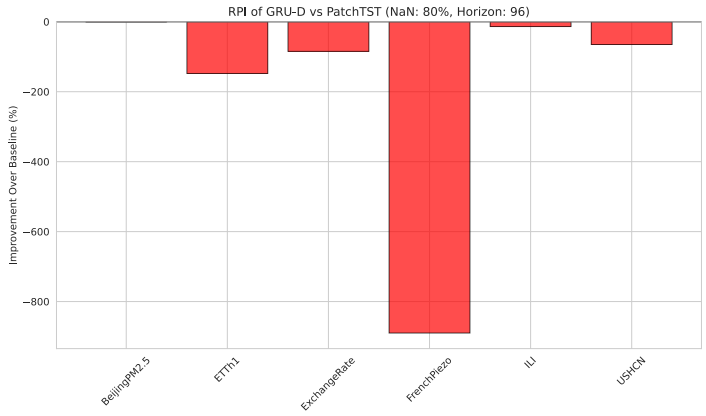


(c)

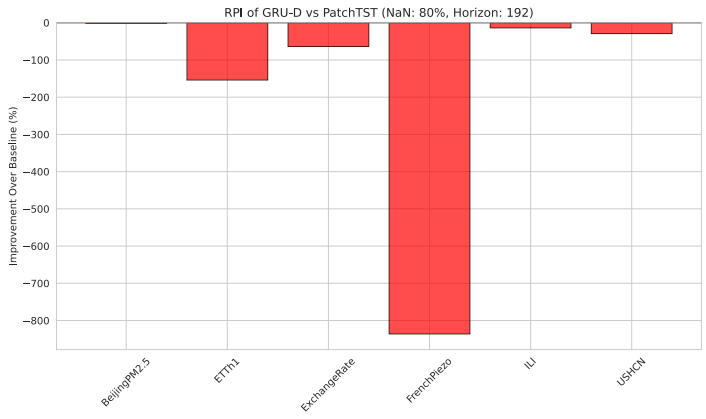


(d)

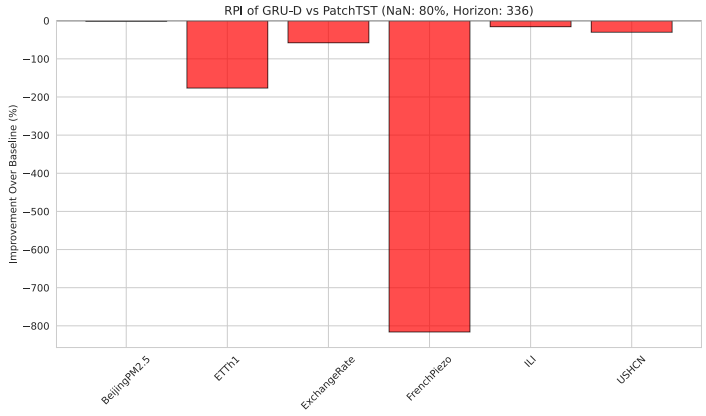
Figure A.55: Relative Performance Improvement of GRU-D over PatchTST with 50% missing data.



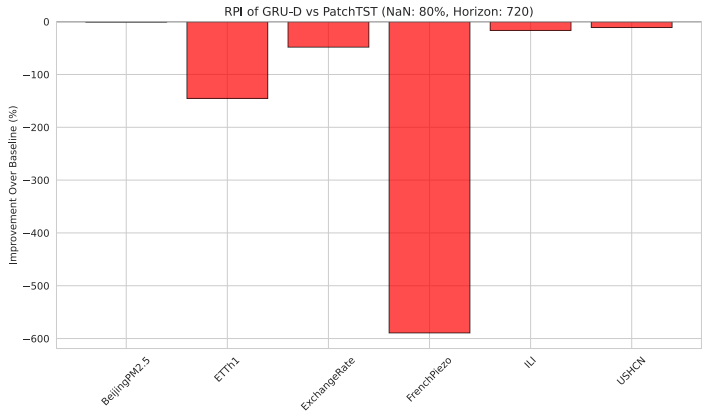
(a)



(b)

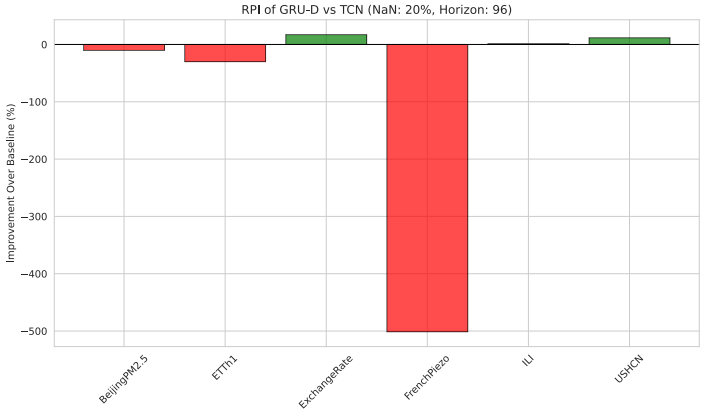


(c)

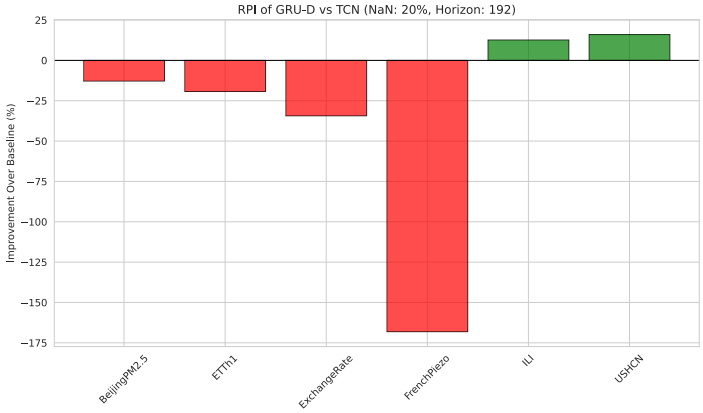


(d)

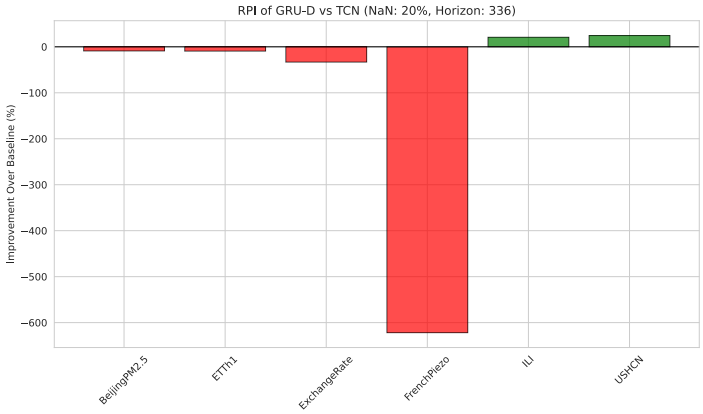
Figure A.56: Relative Performance Improvement of GRU-D over PatchTST with 80% missing data.



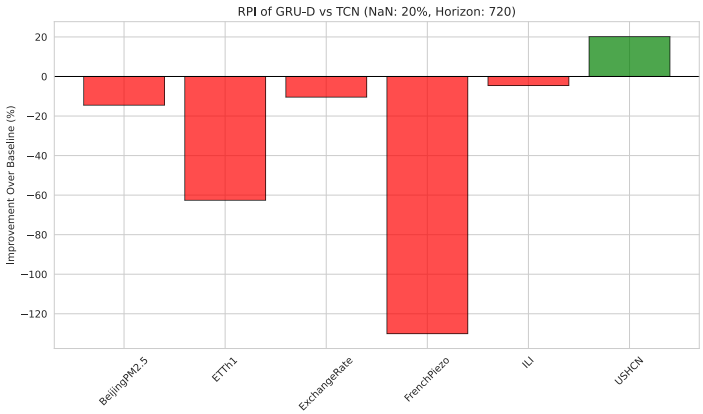
(a)



(b)

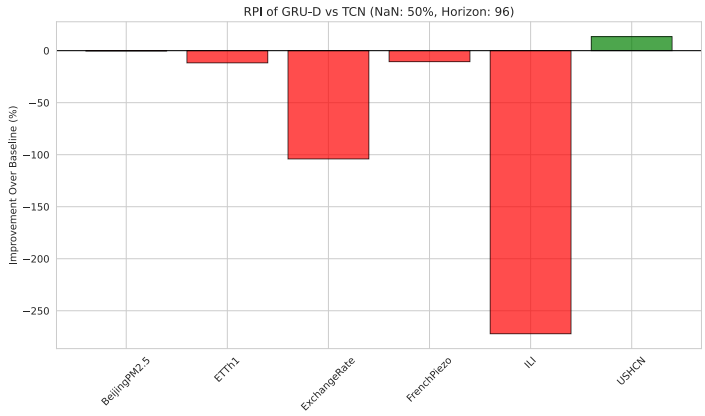


(c)

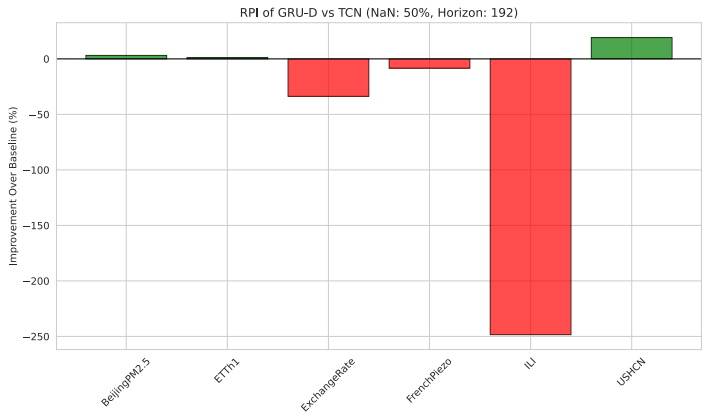


(d)

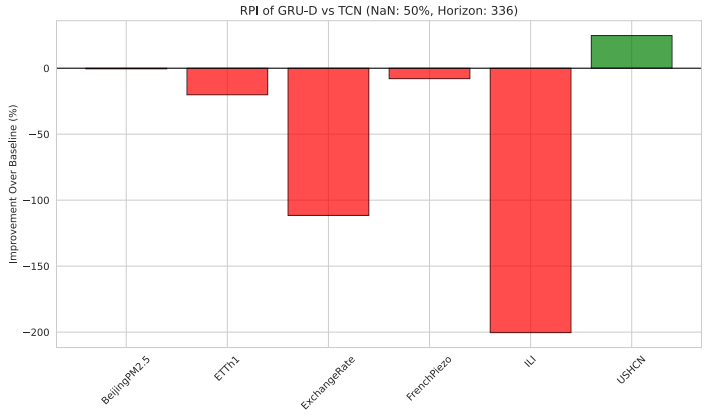
Figure A.57: Relative Performance Improvement of GRU-D over TCN with 20% missing data.



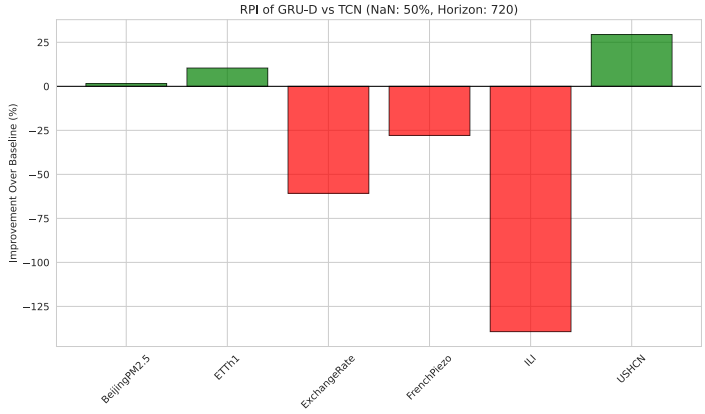
(a)



(b)

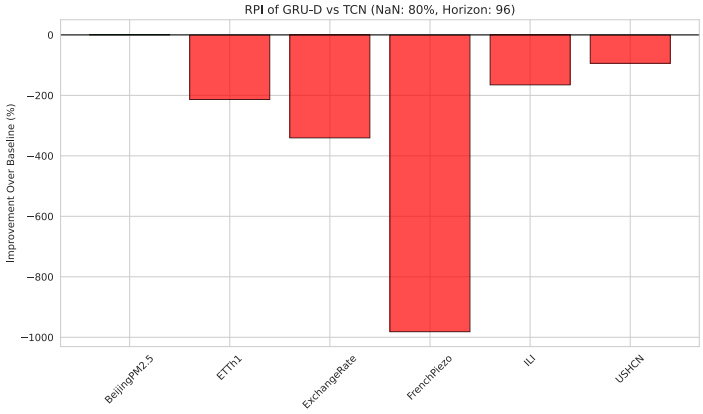


(c)

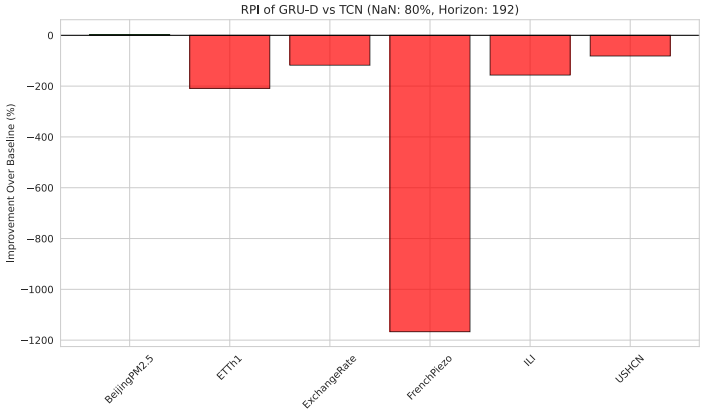


(d)

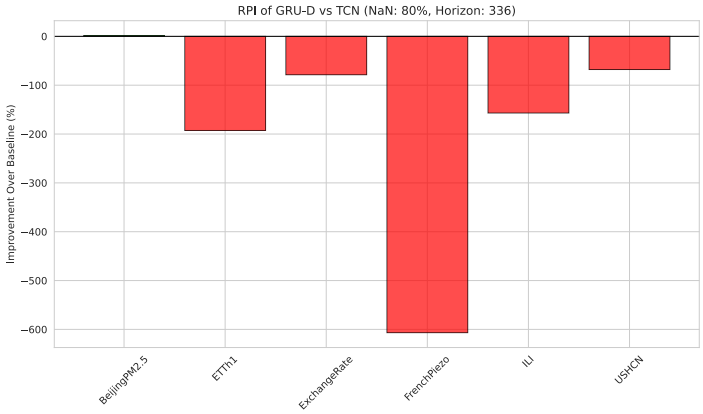
Figure A.58: Relative Performance Improvement of GRU-D over TCN with 50% missing data.



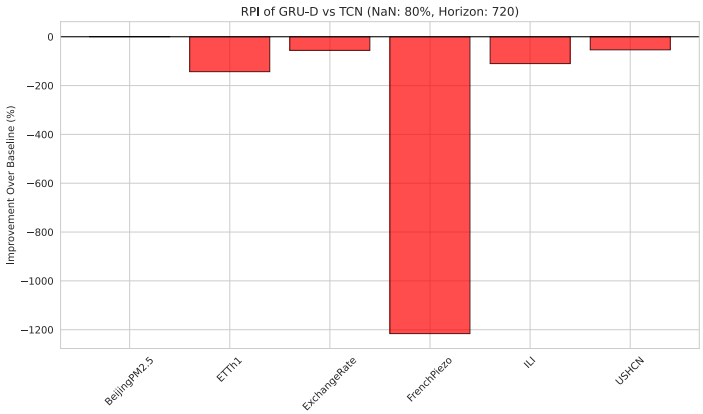
(a)



(b)

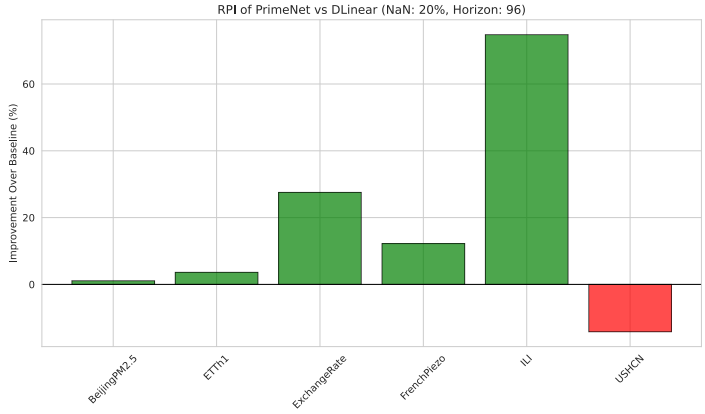


(c)

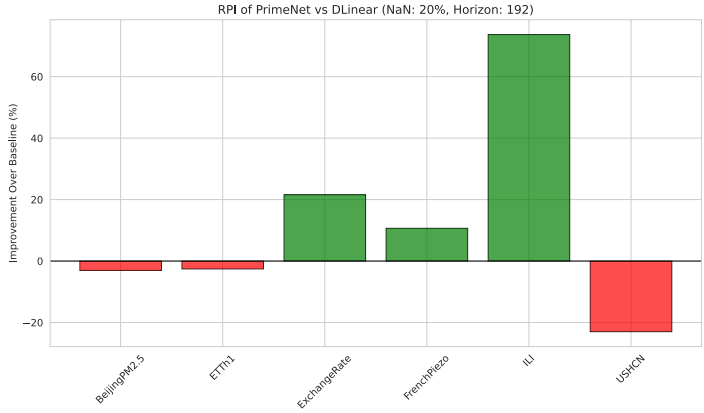


(d)

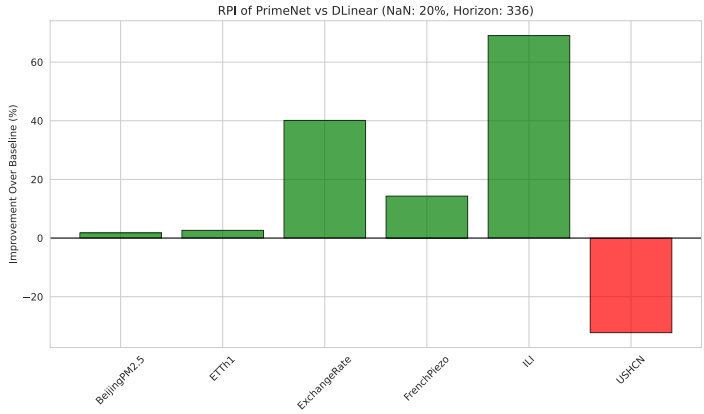
Figure A.59: Relative Performance Improvement of GRU-D over TCN with 80% missing data.



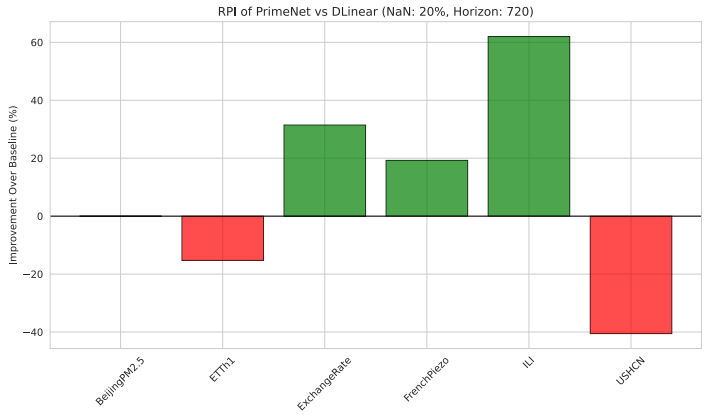
(a)



(b)

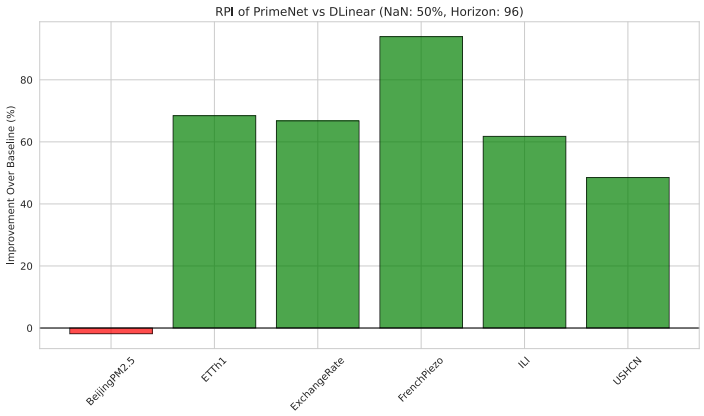


(c)

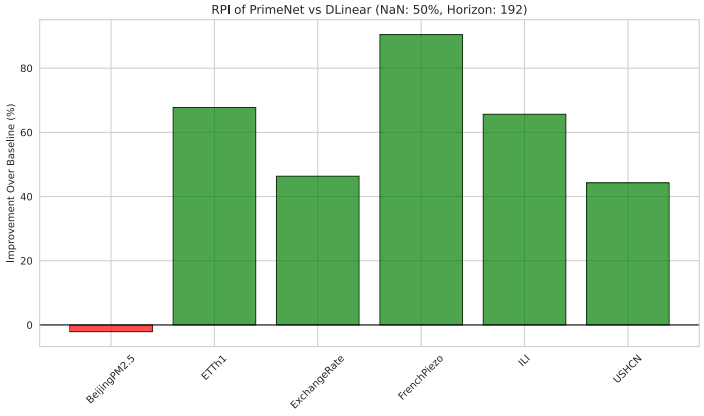


(d)

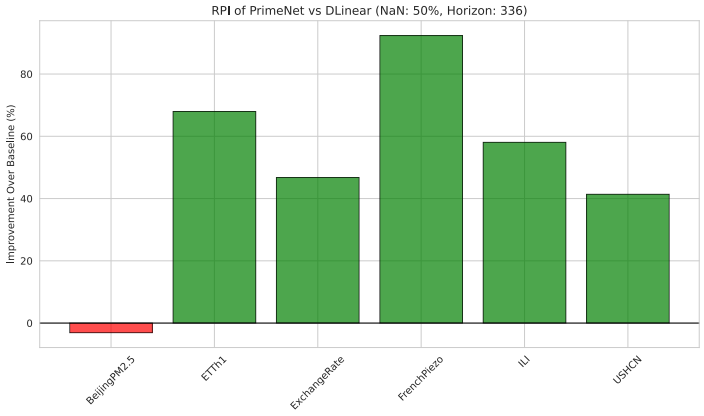
Figure A.60: Relative Performance Improvement of PrimeNet over DLinear with 20% missing data.



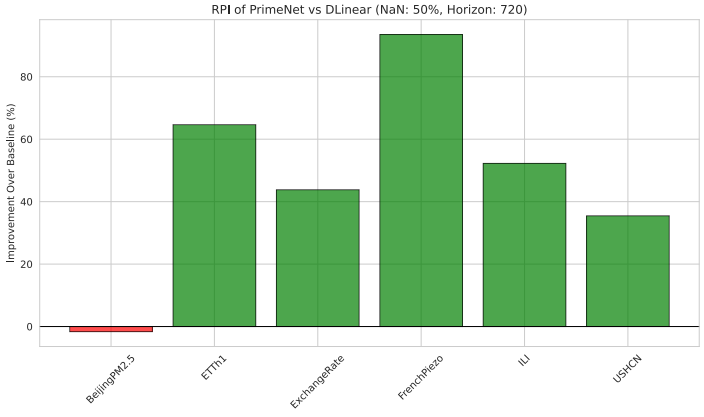
(a)



(b)

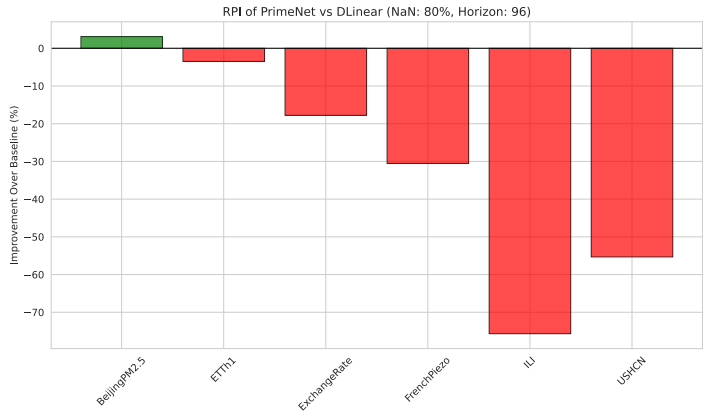


(c)

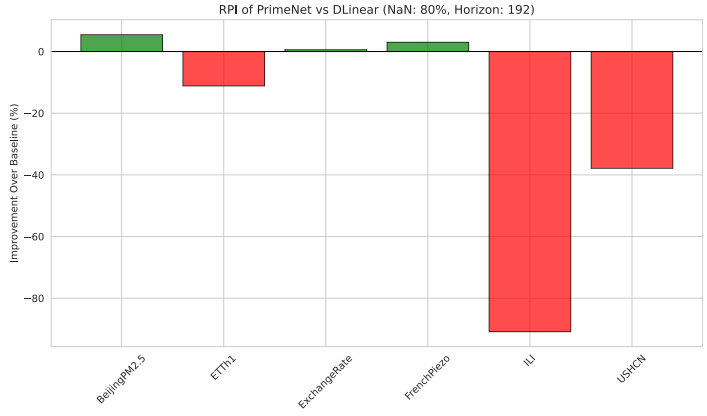


(d)

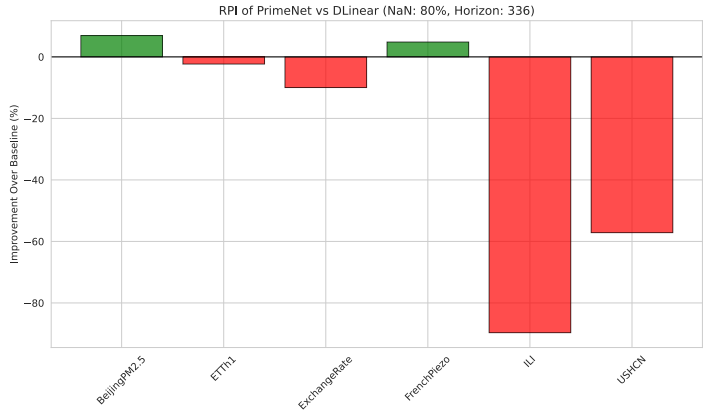
Figure A.61: Relative Performance Improvement of PrimeNet over DLinear with 50% missing data.



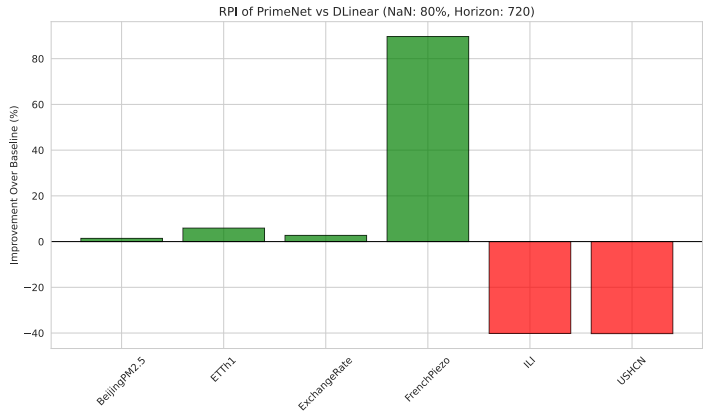
(a)



(b)

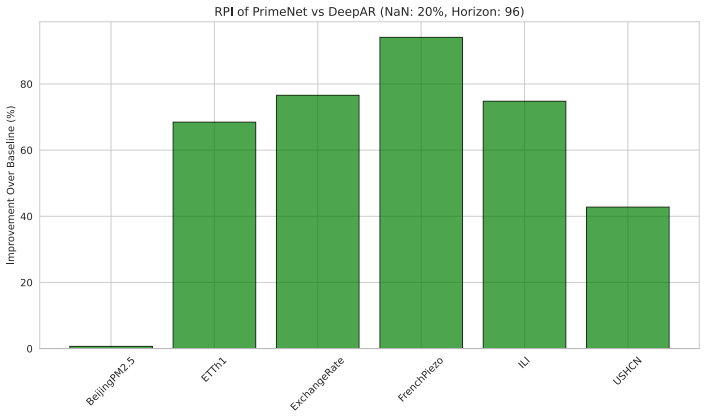


(c)

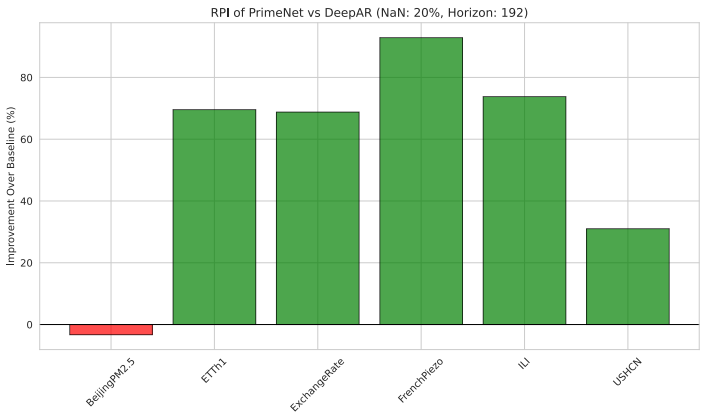


(d)

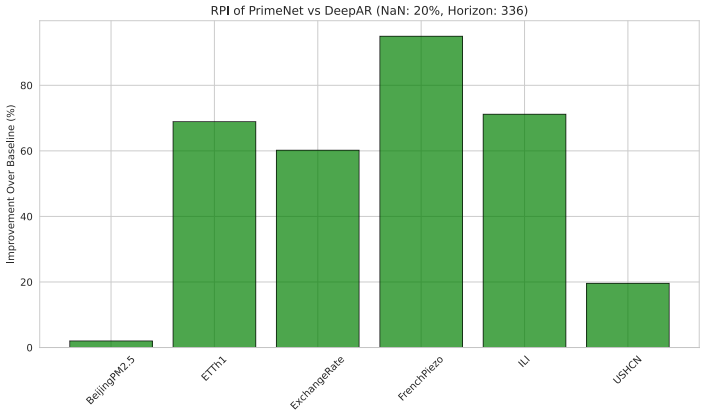
Figure A.62: Relative Performance Improvement of PrimeNet over DLinear with 80% missing data.



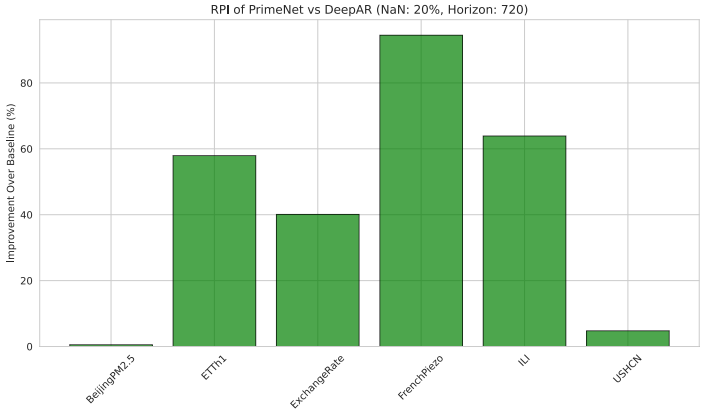
(a)



(b)

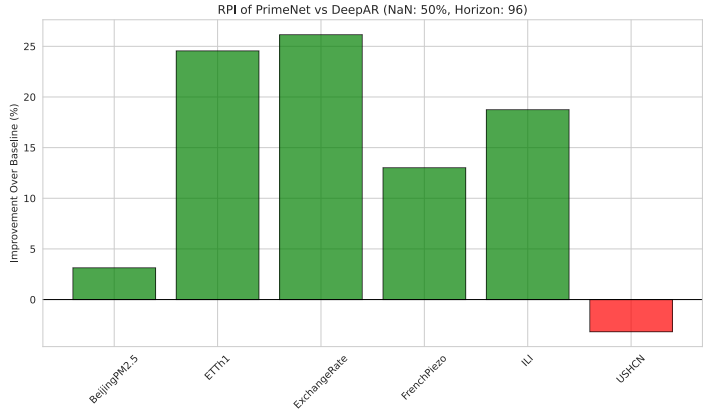


(c)

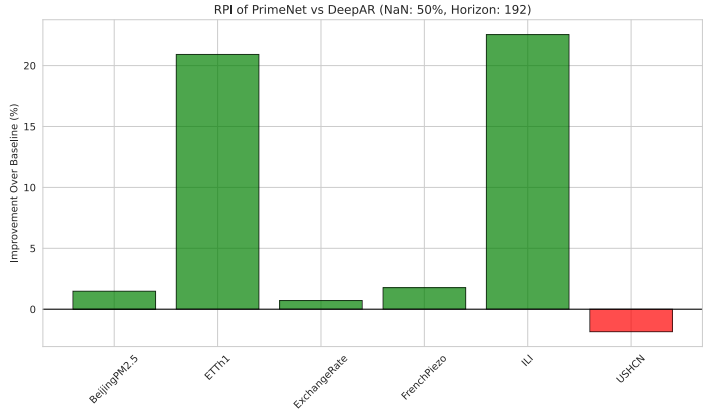


(d)

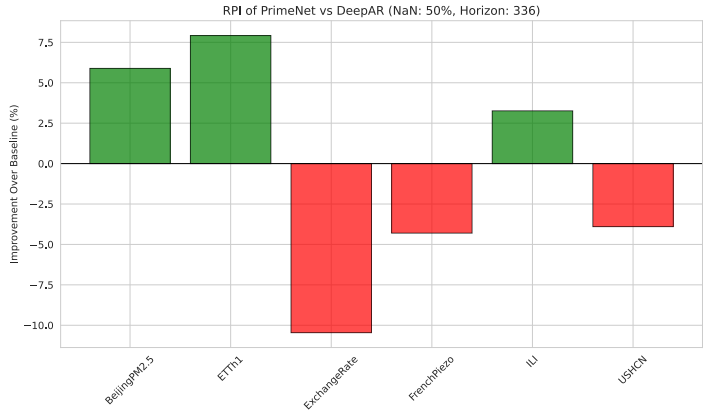
Figure A.63: Relative Performance Improvement of PrimeNet over DeepAR with 20% missing data.



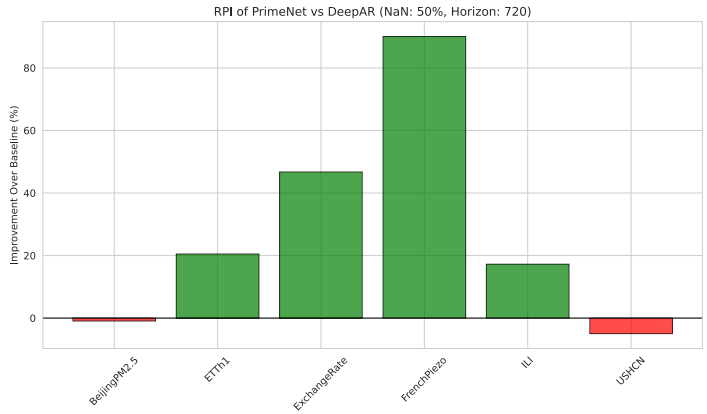
(a)



(b)

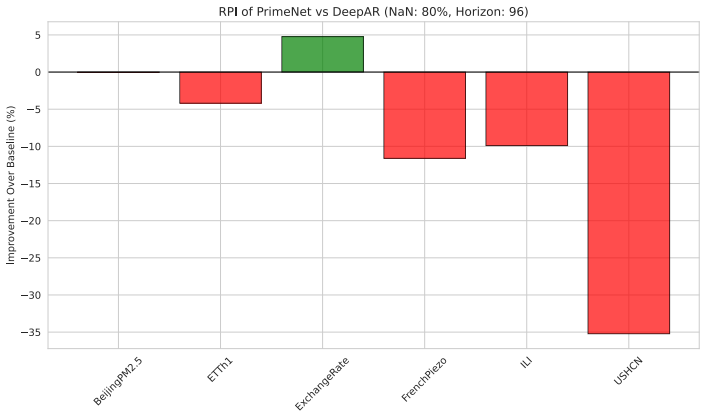


(c)

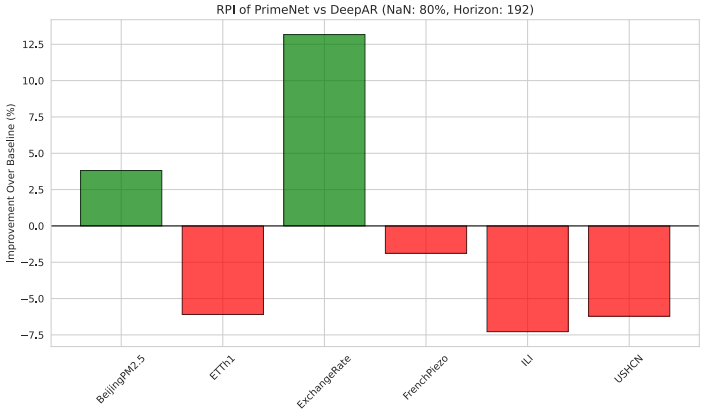


(d)

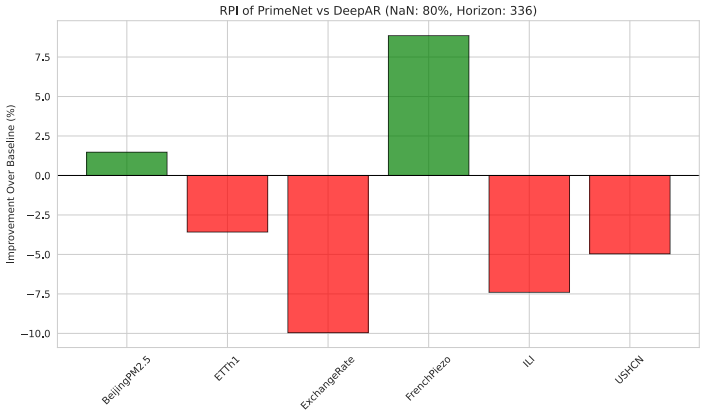
Figure A.64: Relative Performance Improvement of PrimeNet over DeepAR with 50% missing data.



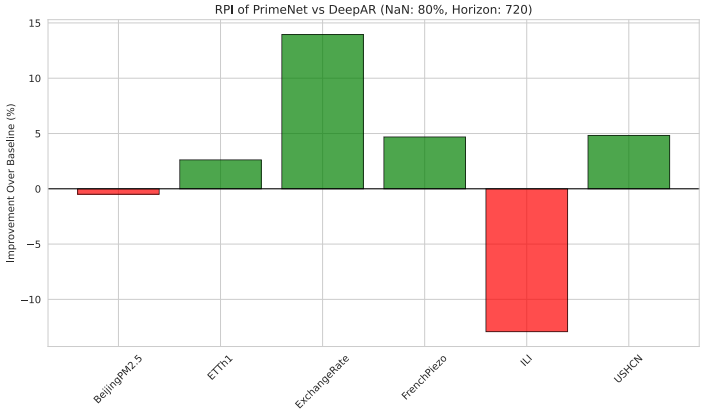
(a)



(b)

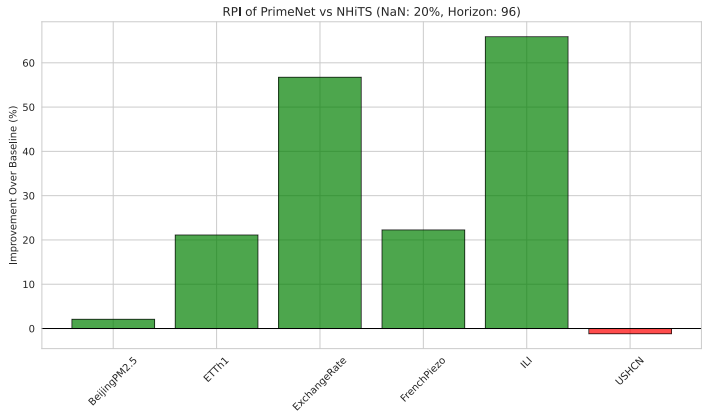


(c)

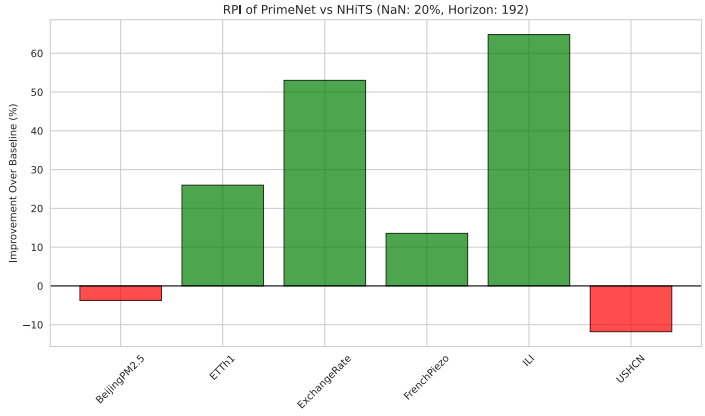


(d)

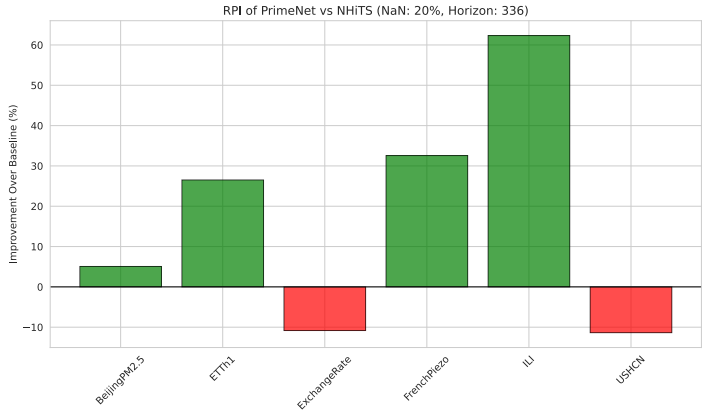
Figure A.65: Relative Performance Improvement of PrimeNet over DeepAR with 80% missing data.



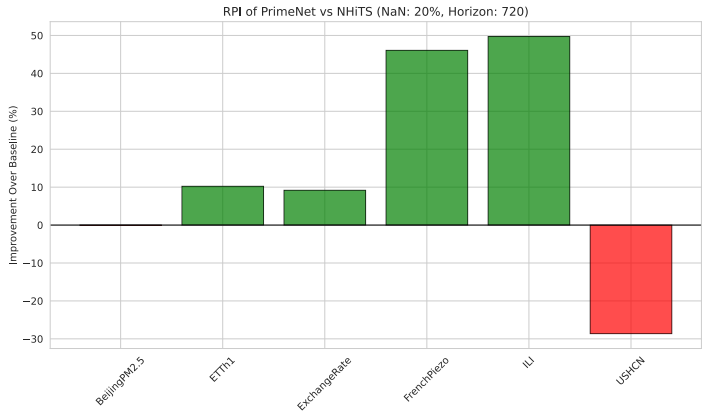
(a)



(b)

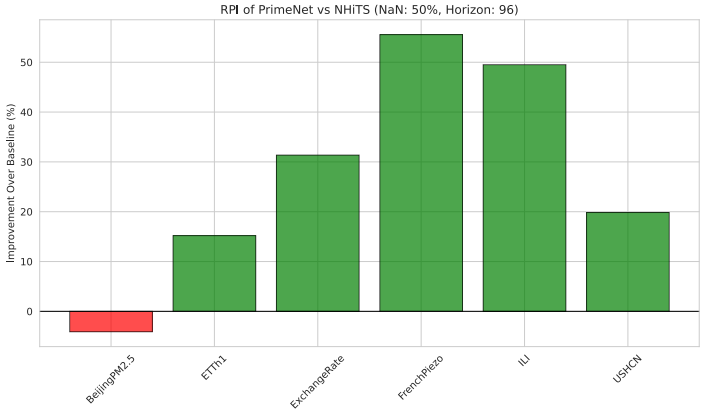


(c)

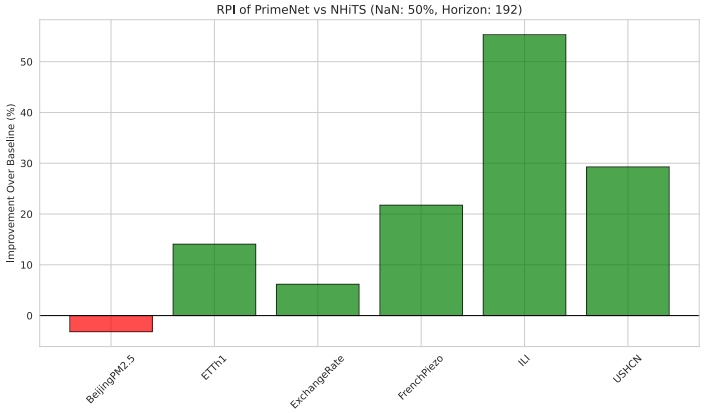


(d)

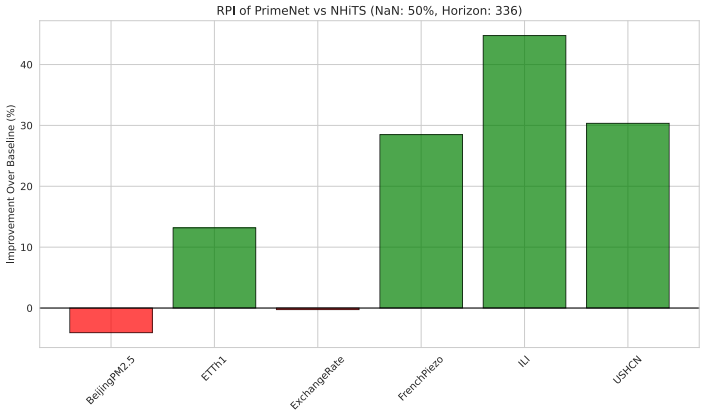
Figure A.66: Relative Performance Improvement of PrimeNet over NHiTS with 20% missing data.



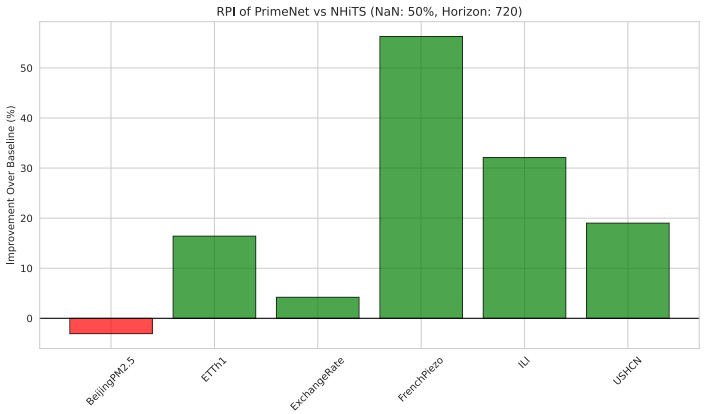
(a)



(b)

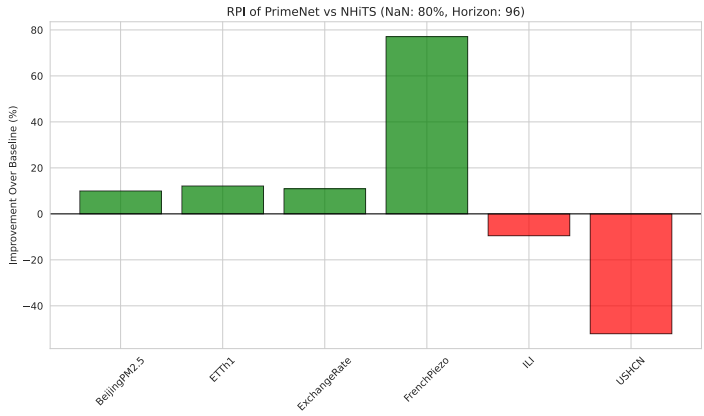


(c)

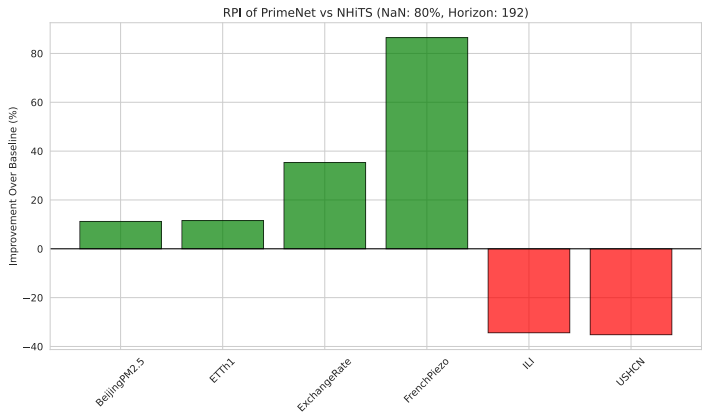


(d)

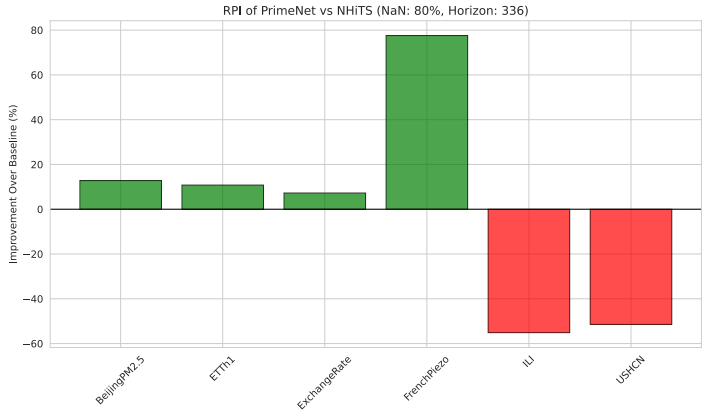
Figure A.67: Relative Performance Improvement of PrimeNet over NHiTS with 50% missing data.



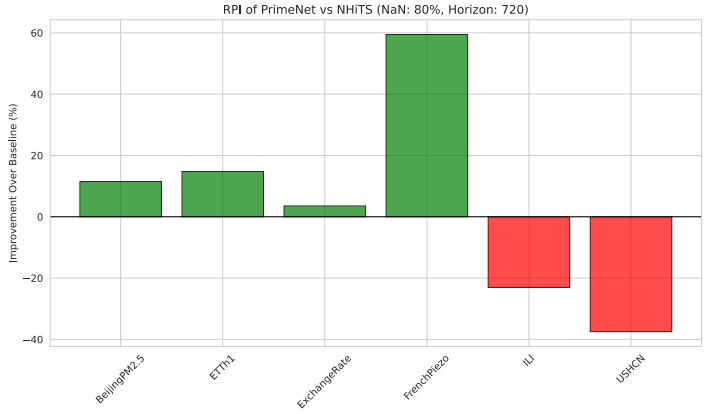
(a)



(b)

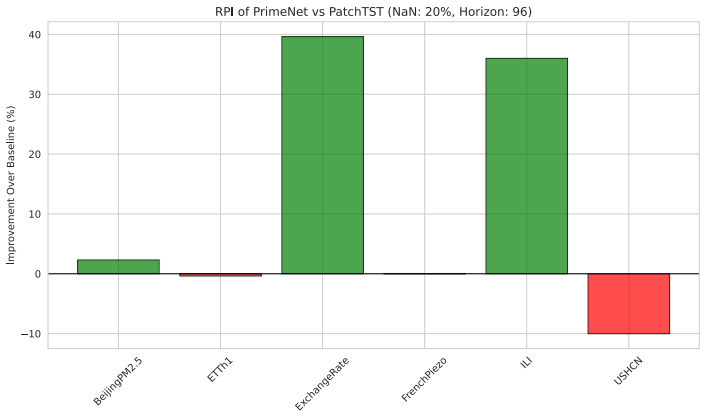


(c)

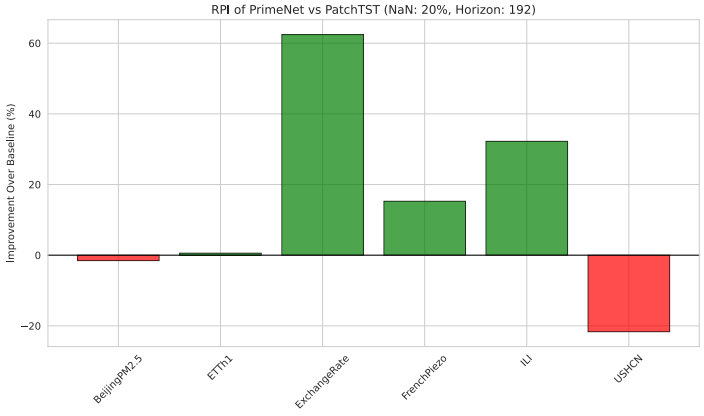


(d)

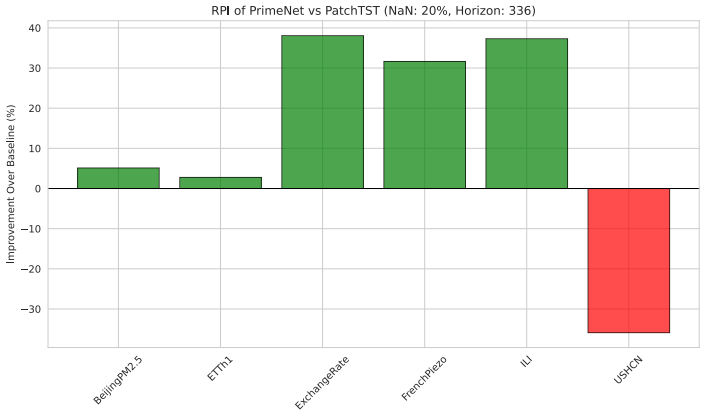
Figure A.68: Relative Performance Improvement of PrimeNet over NHiTS with 80% missing data.



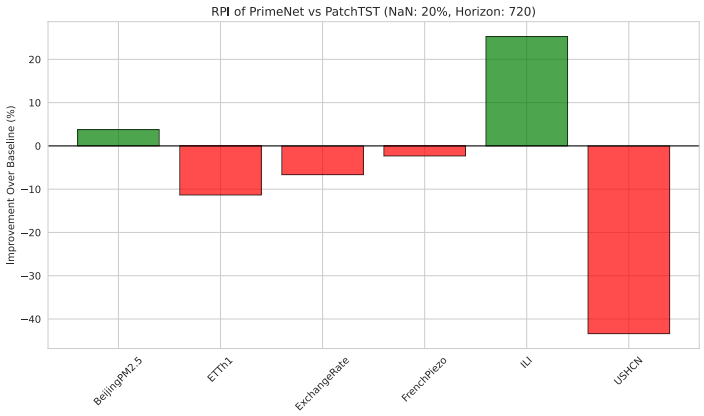
(a)



(b)

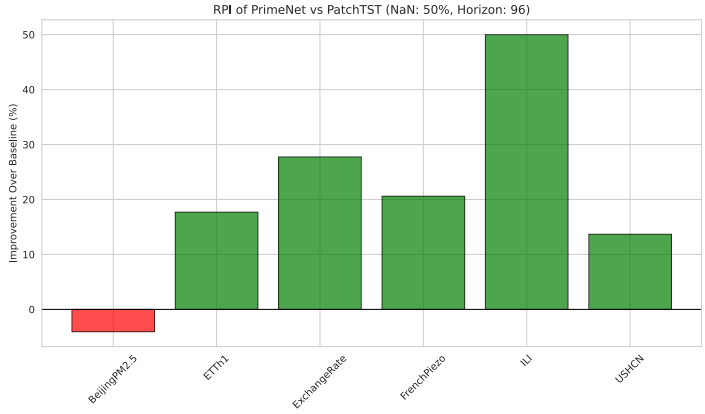


(c)

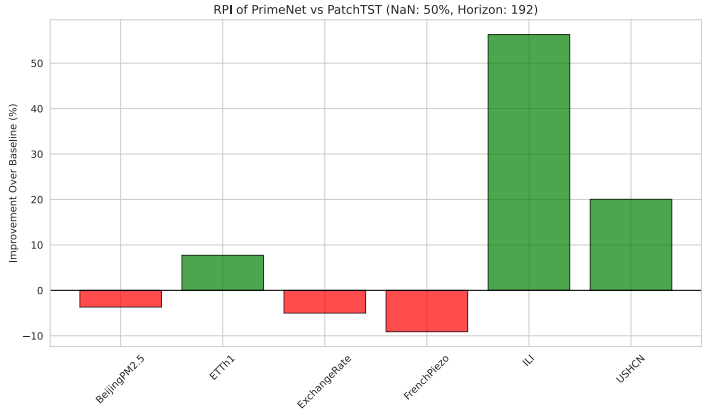


(d)

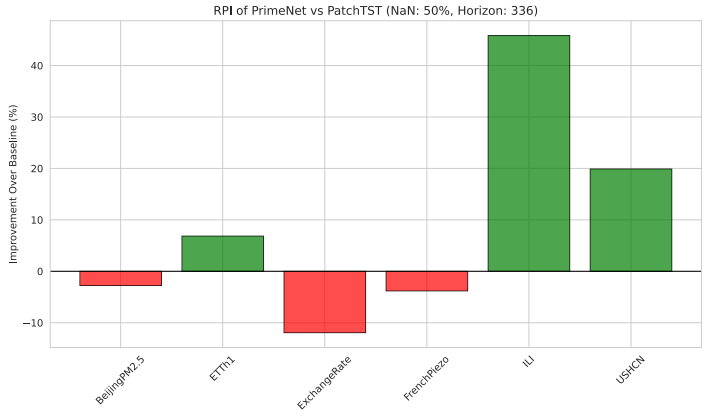
Figure A.69: Relative Performance Improvement of PrimeNet over PatchTST with 20% missing data.



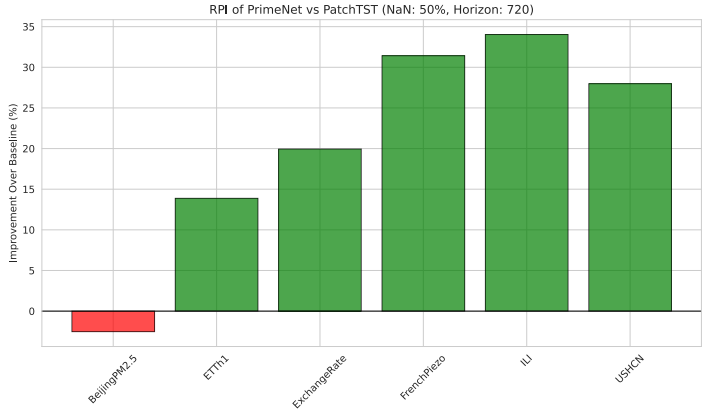
(a)



(b)

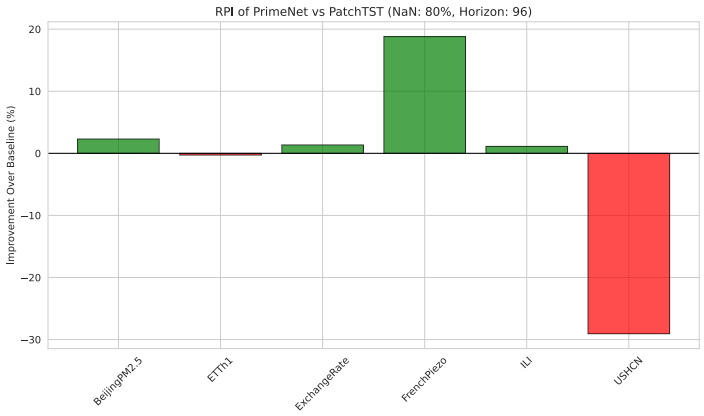


(c)

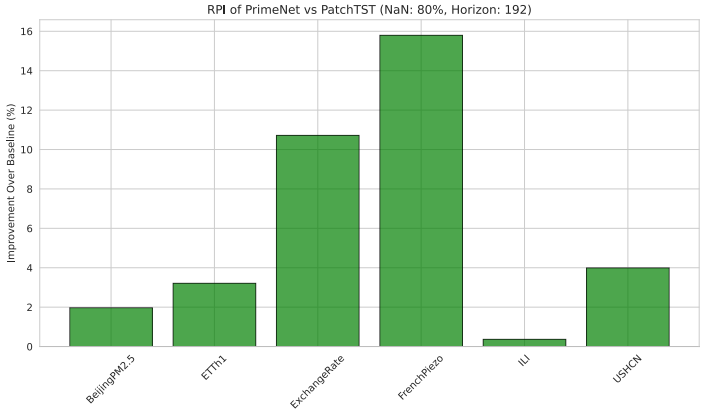


(d)

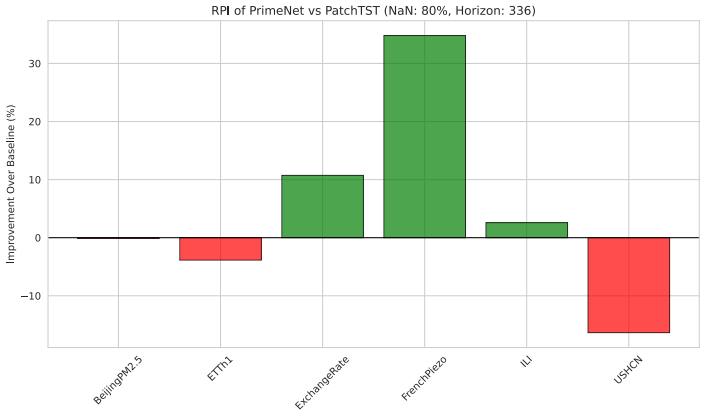
Figure A.70: Relative Performance Improvement of PrimeNet over PatchTST with 50% missing data.



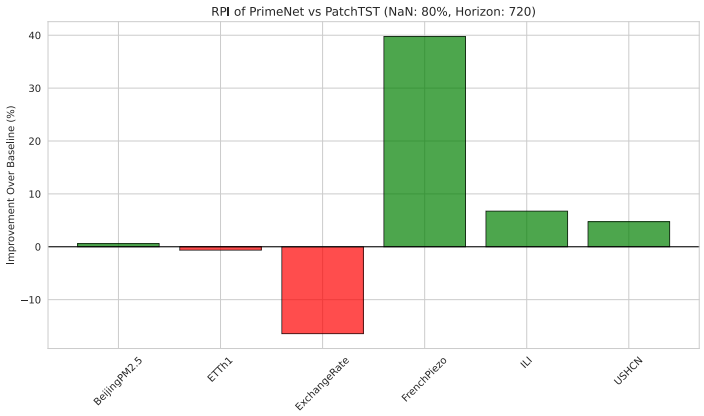
(a)



(b)

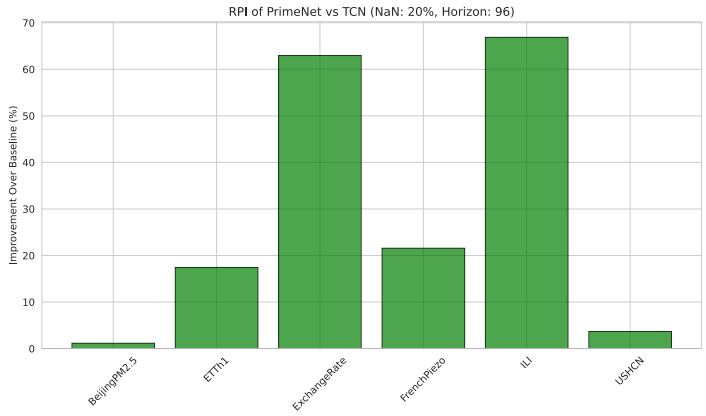


(c)

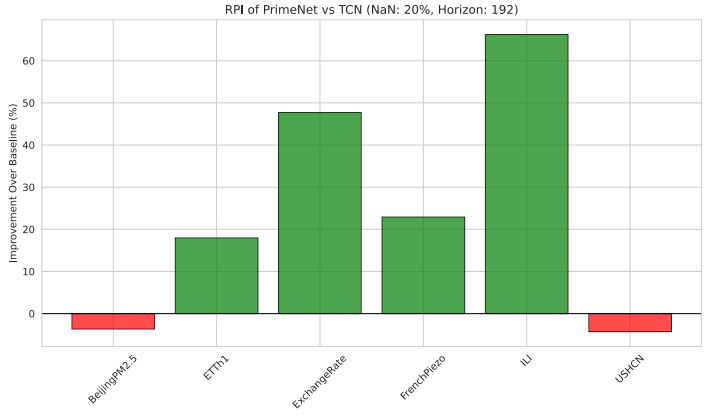


(d)

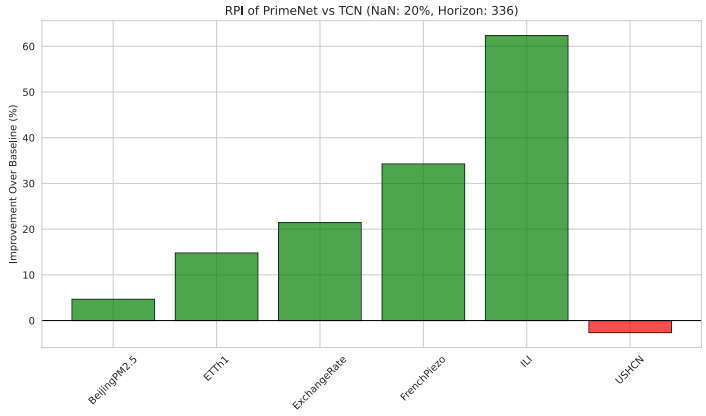
Figure A.71: Relative Performance Improvement of PrimeNet over PatchTST with 80% missing data.



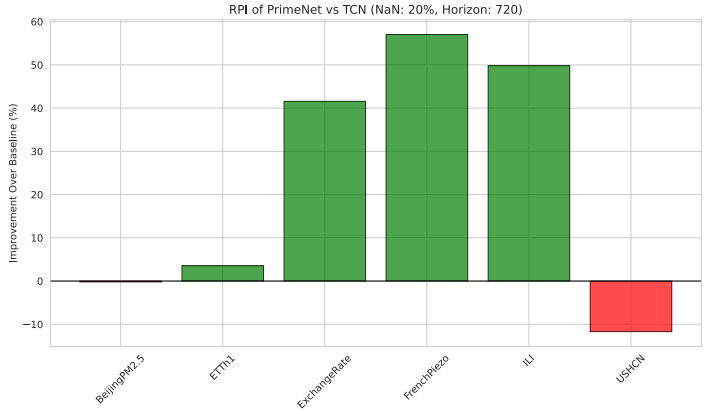
(a)



(b)

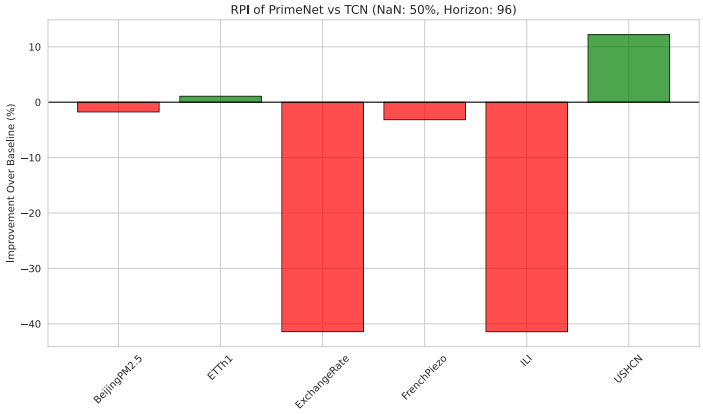


(c)

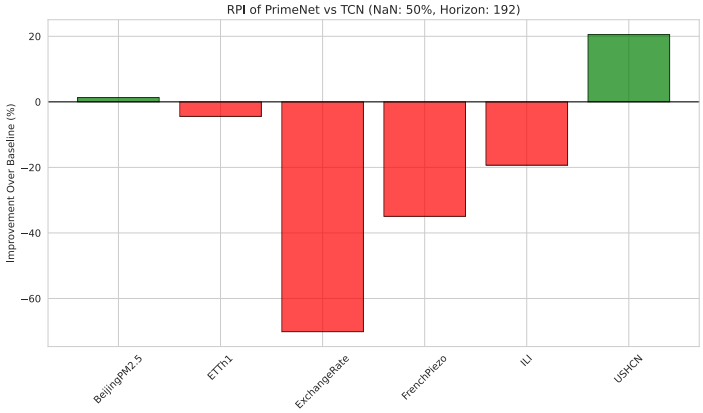


(d)

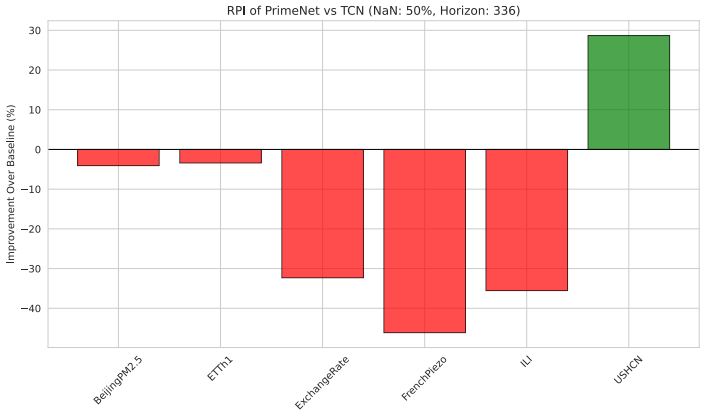
Figure A.72: Relative Performance Improvement of PrimeNet over TCN with 20% missing data.



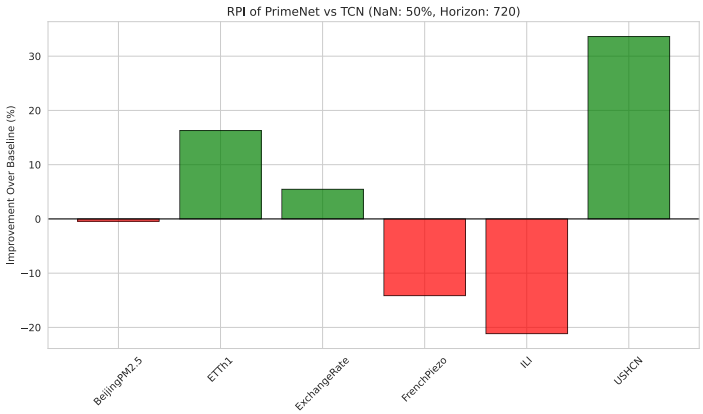
(a)



(b)

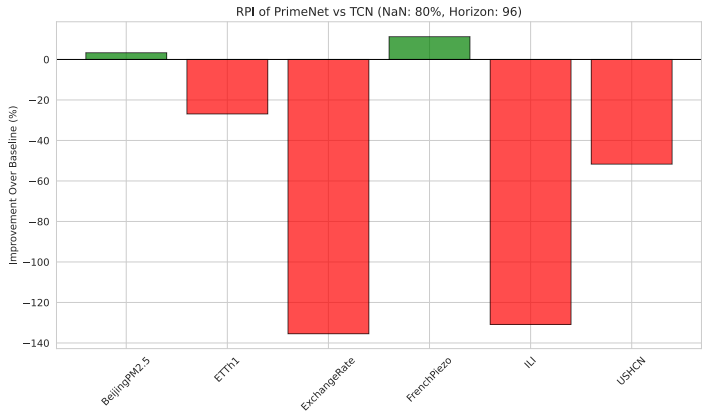


(c)

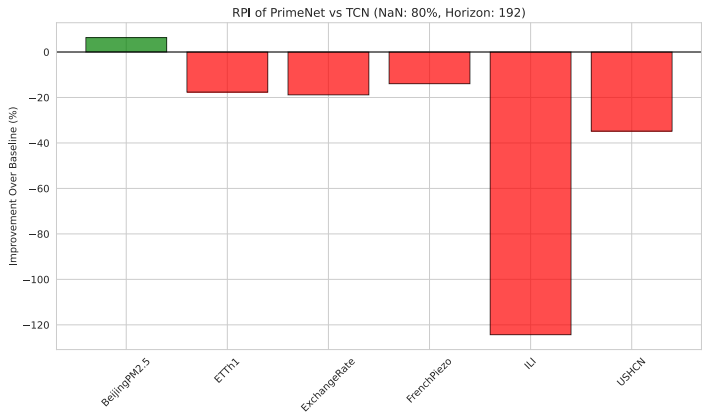


(d)

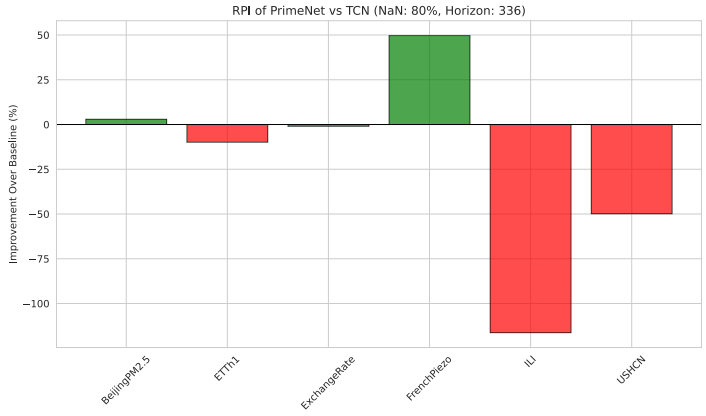
Figure A.73: Relative Performance Improvement of PrimeNet over TCN with 50% missing data.



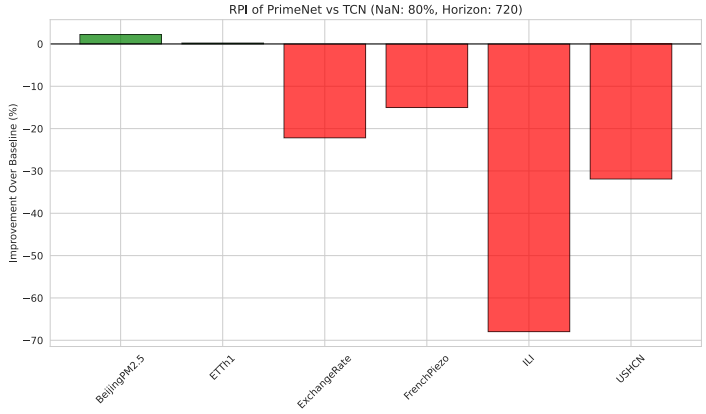
(a)



(b)

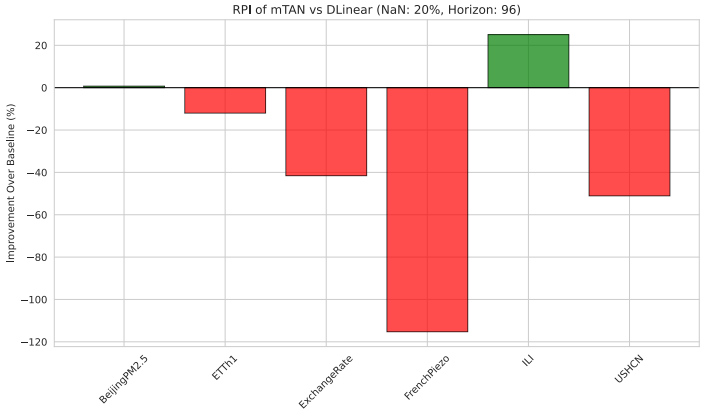


(c)

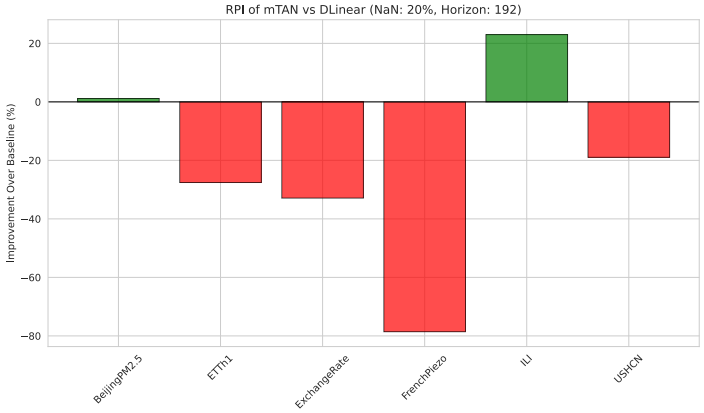


(d)

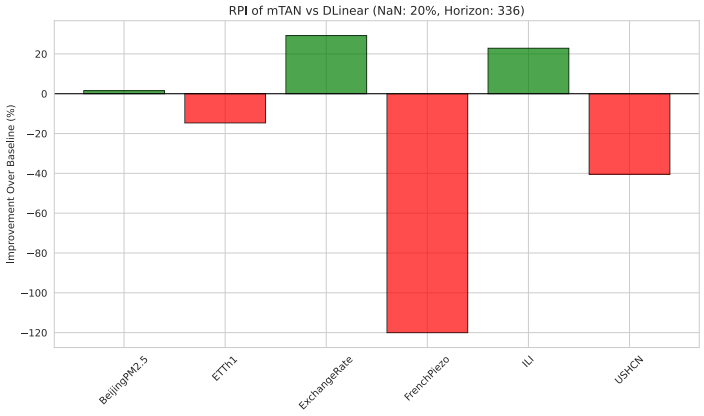
Figure A.74: Relative Performance Improvement of PrimeNet over TCN with 80% missing data.



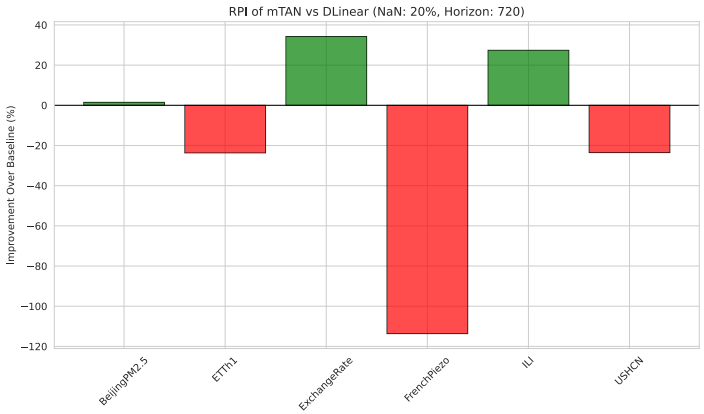
(a)



(b)

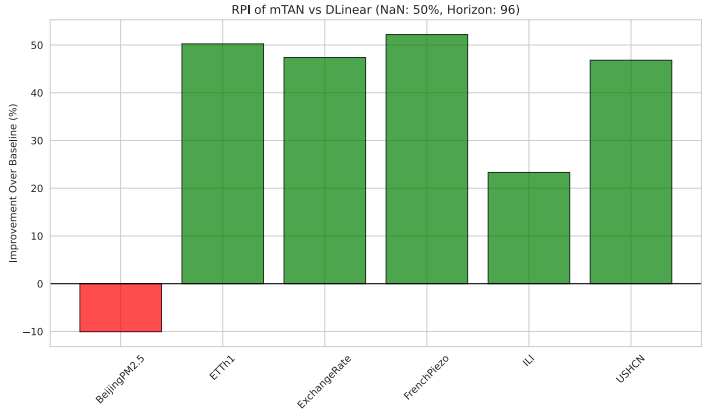


(c)

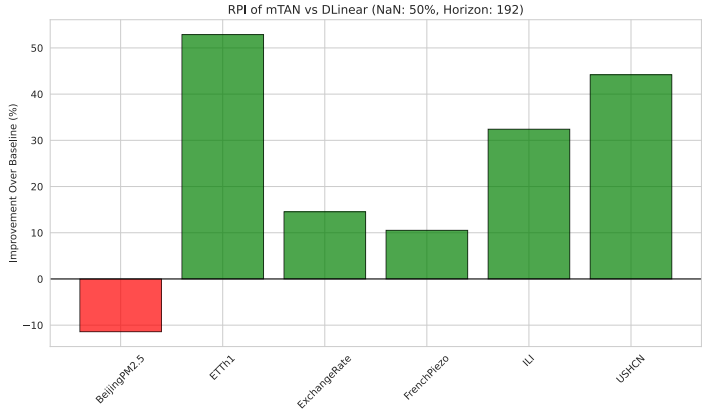


(d)

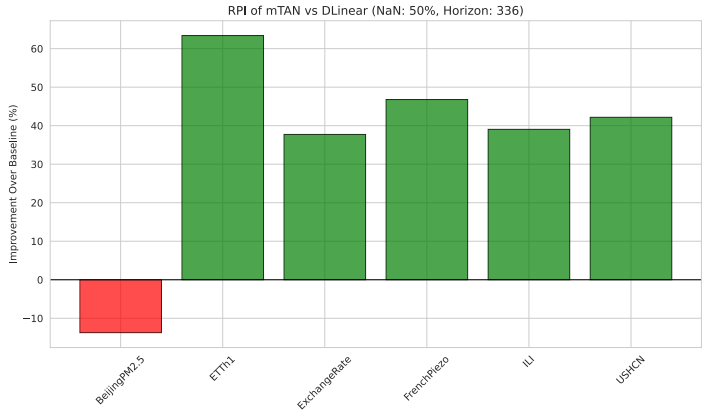
Figure A.75: Relative Performance Improvement of mTAN over DLinear with 20% missing data.



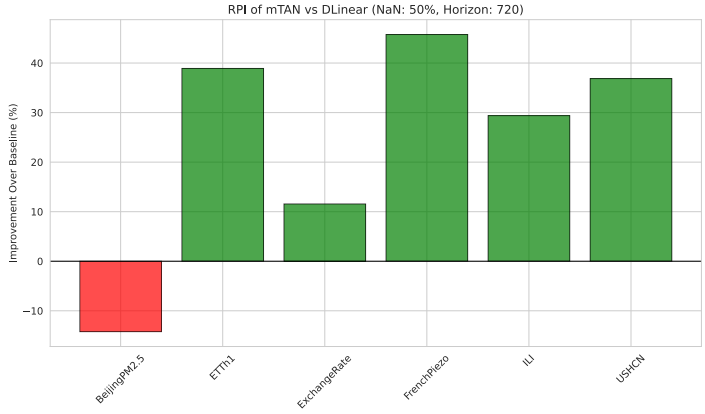
(a)



(b)

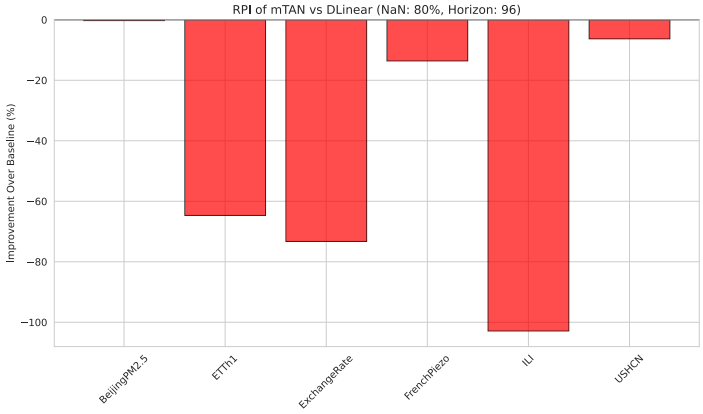


(c)

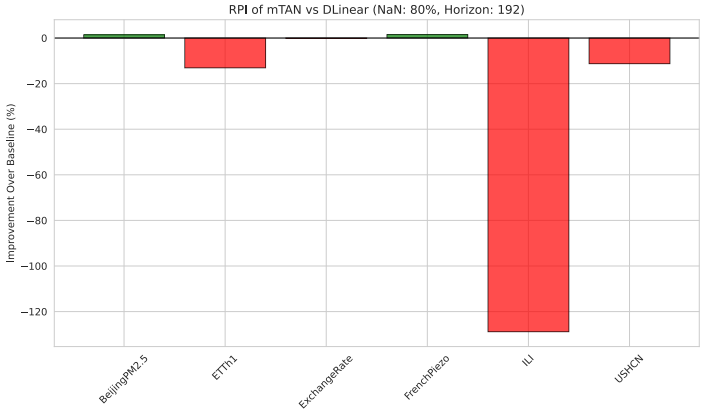


(d)

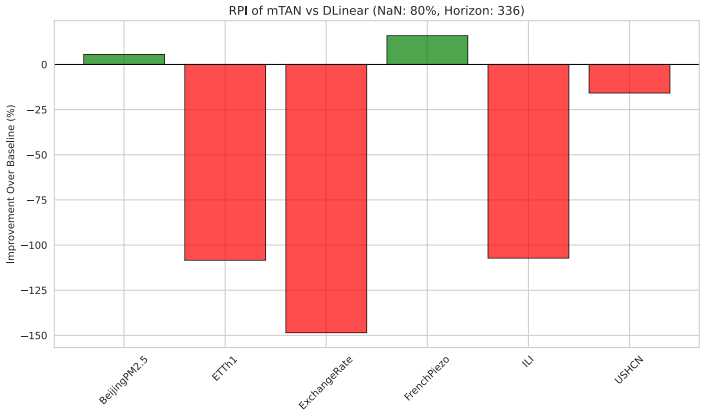
Figure A.76: Relative Performance Improvement of mTAN over DLinear with 50% missing data.



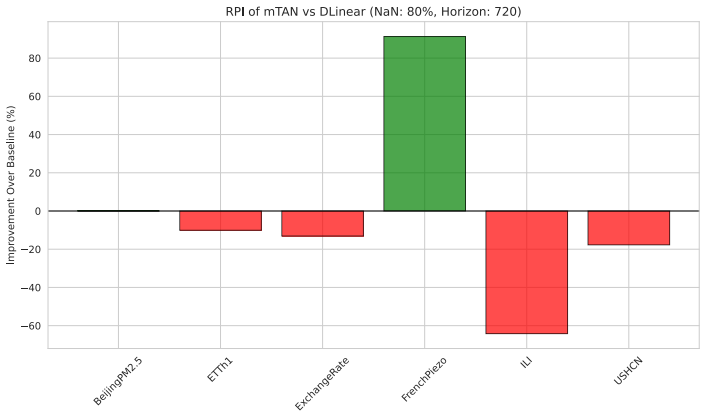
(a)



(b)

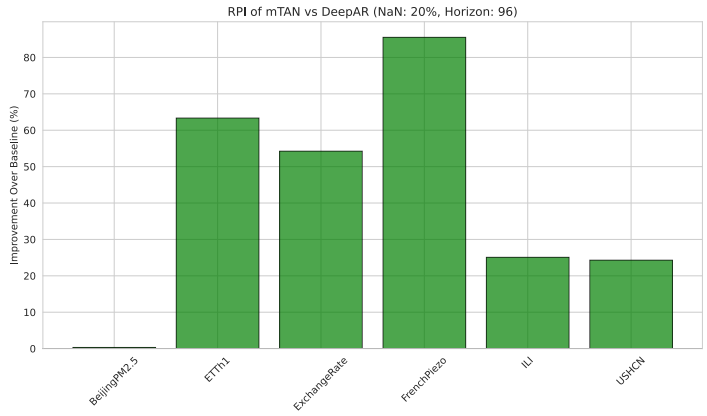


(c)

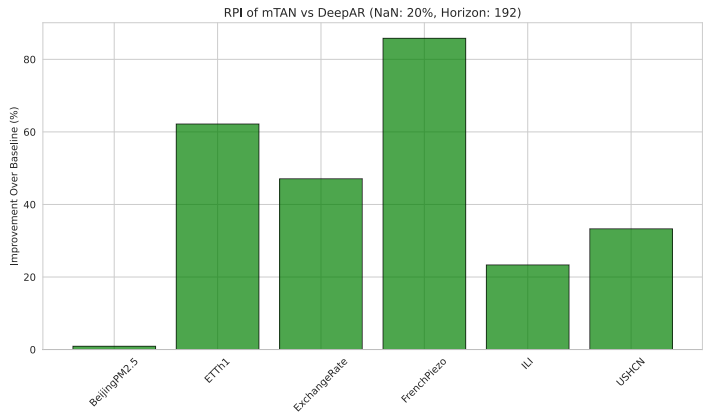


(d)

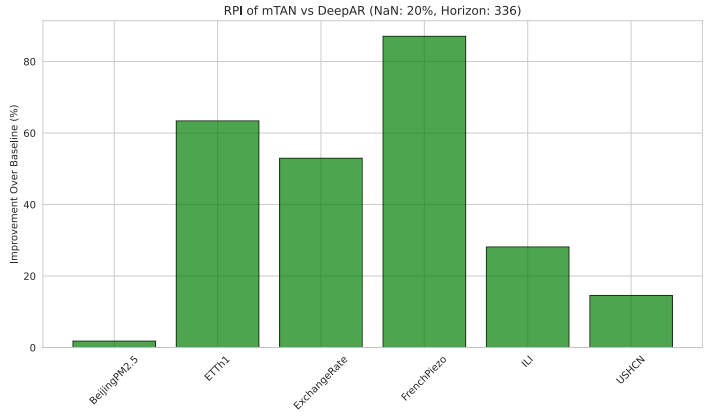
Figure A.77: Relative Performance Improvement of mTAN over DLinear with 50% missing data.



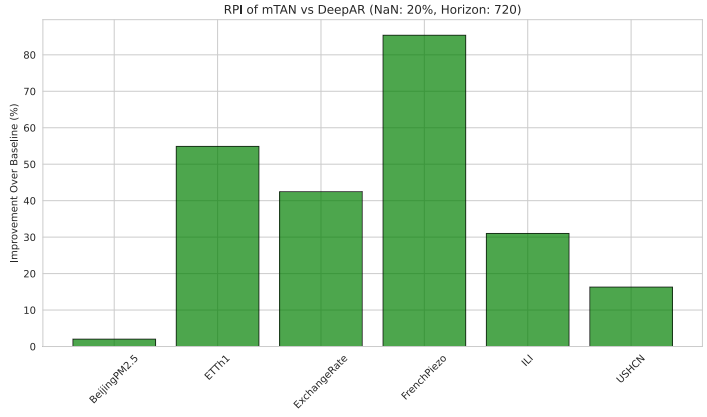
(a)



(b)

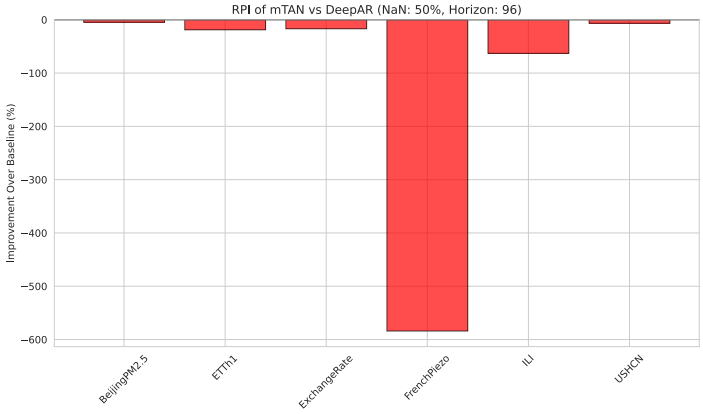


(c)

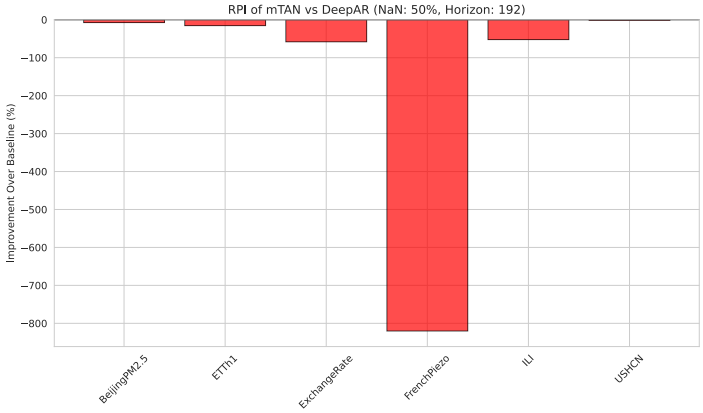


(d)

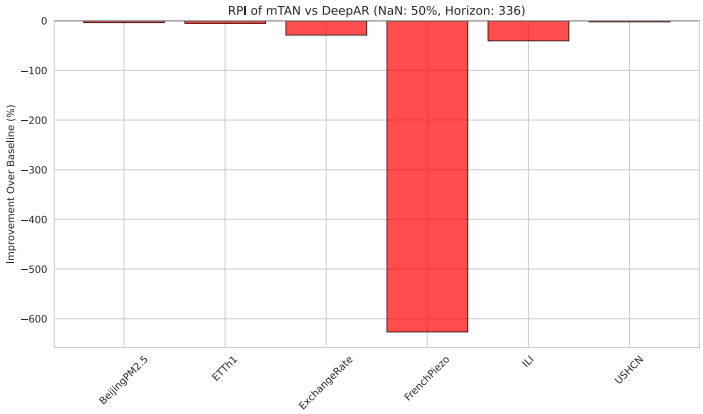
Figure A.78: Relative Performance Improvement of mTAN over DeepAR with 20% missing data.



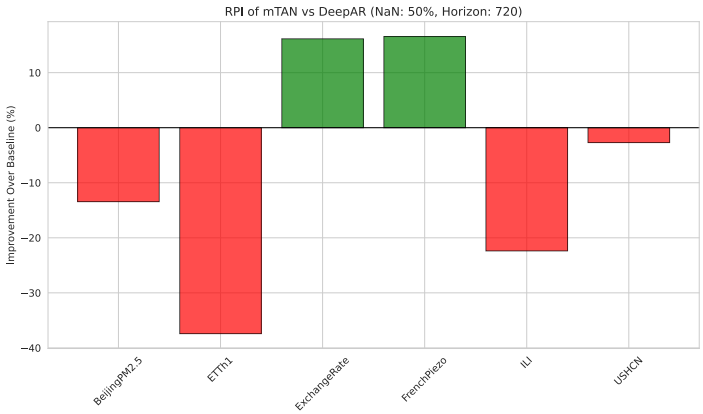
(a)



(b)

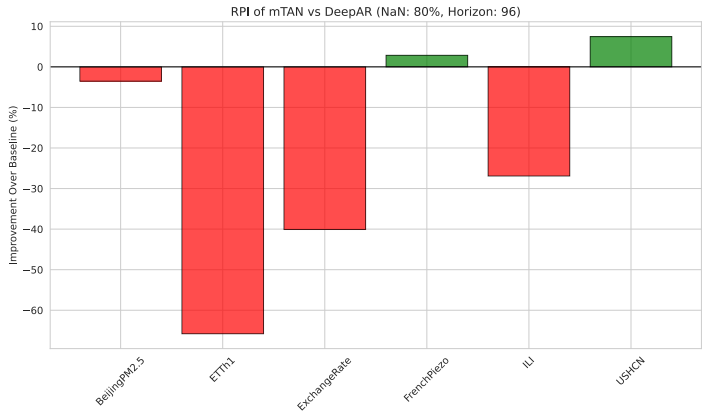


(c)

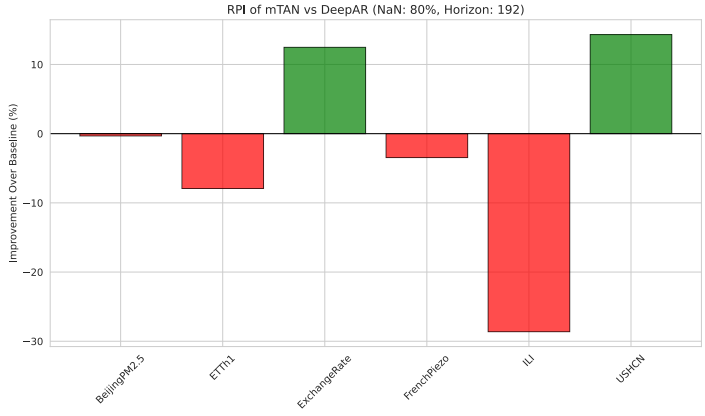


(d)

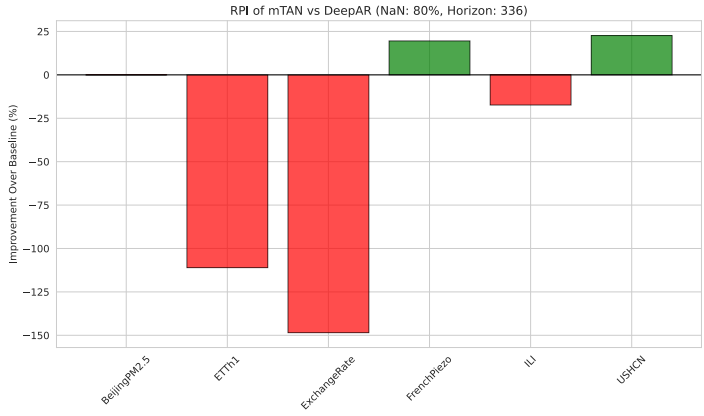
Figure A.79: Relative Performance Improvement of mTAN over DeepAR with 50% missing data.



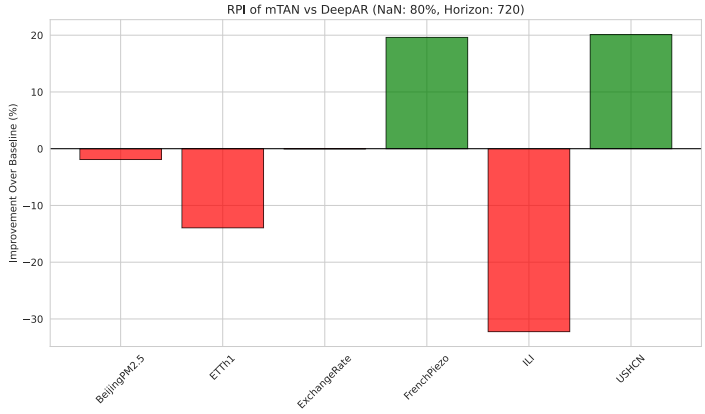
(a)



(b)

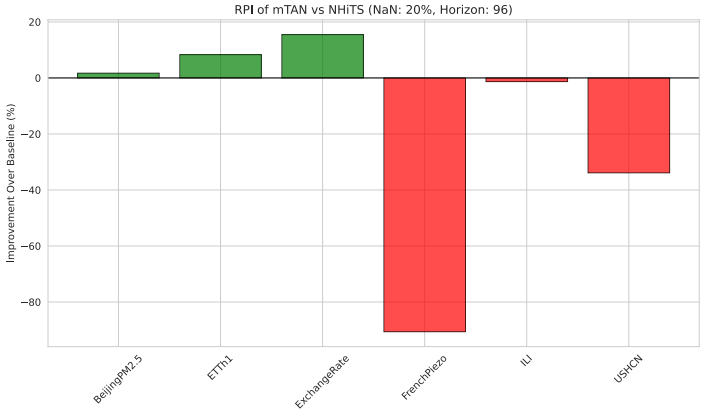


(c)

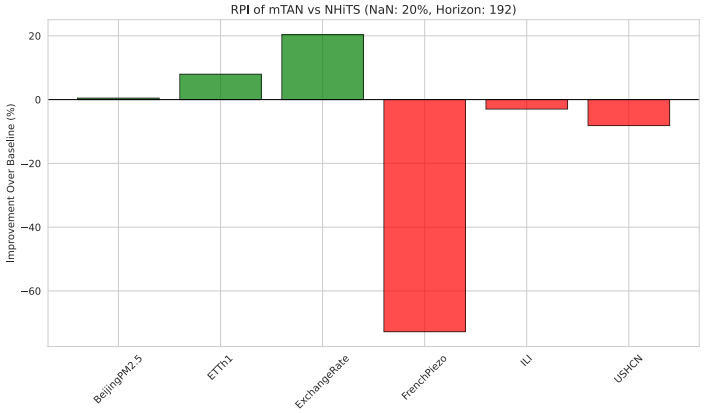


(d)

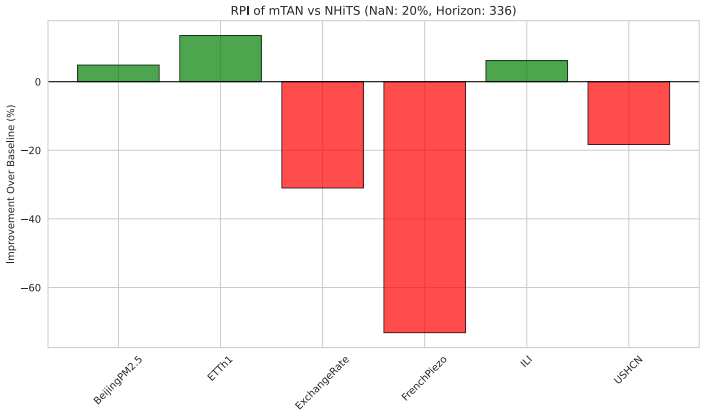
Figure A.80: Relative Performance Improvement of mTAN over DeepAR with 80% missing data.



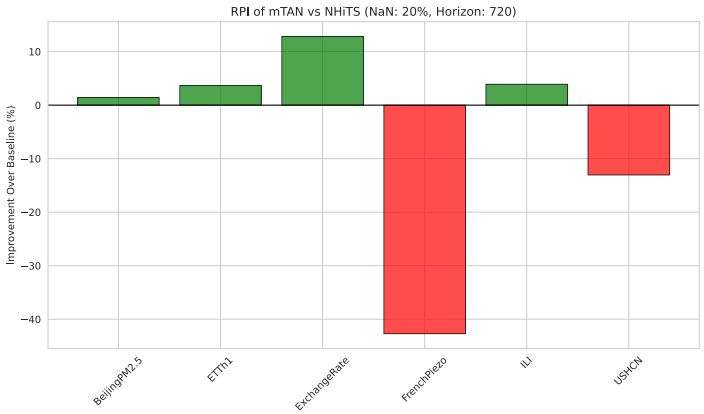
(a)



(b)

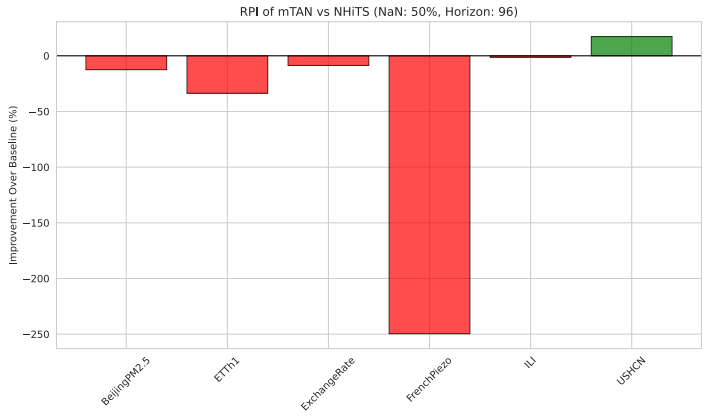


(c)

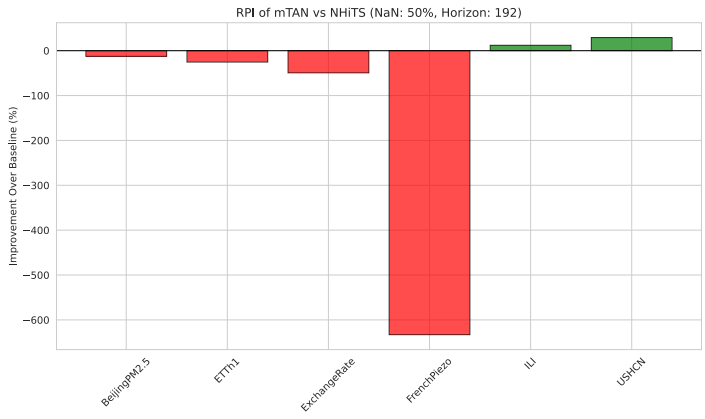


(d)

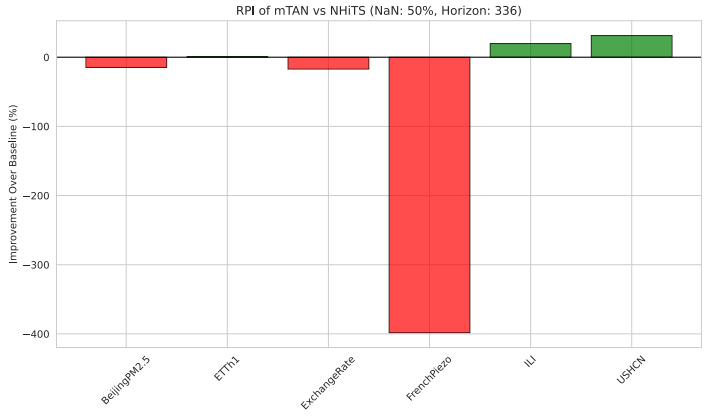
Figure A.81: Relative Performance Improvement of mTAN over NHiTS with 20% missing data.



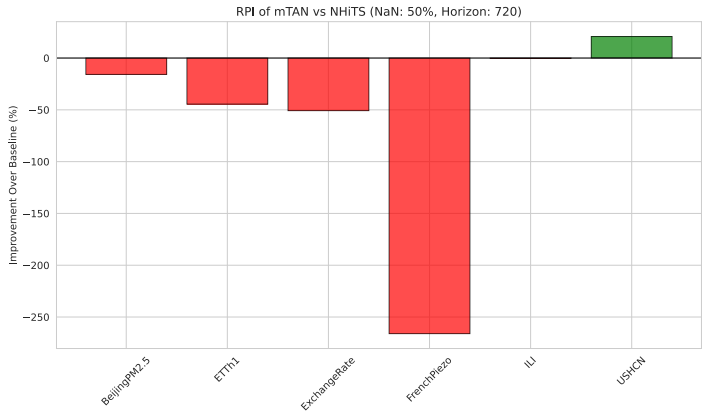
(a)



(b)

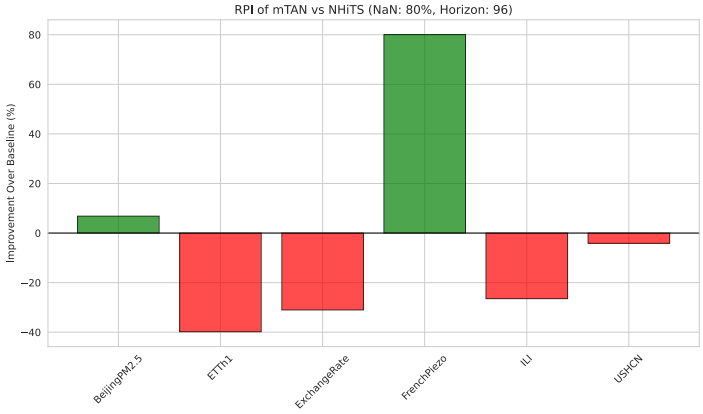


(c)

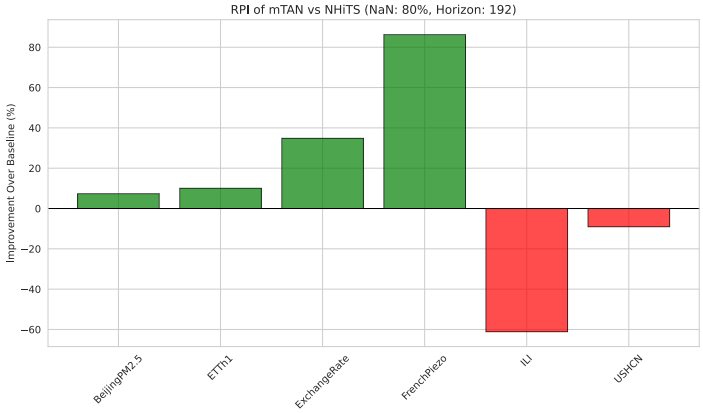


(d)

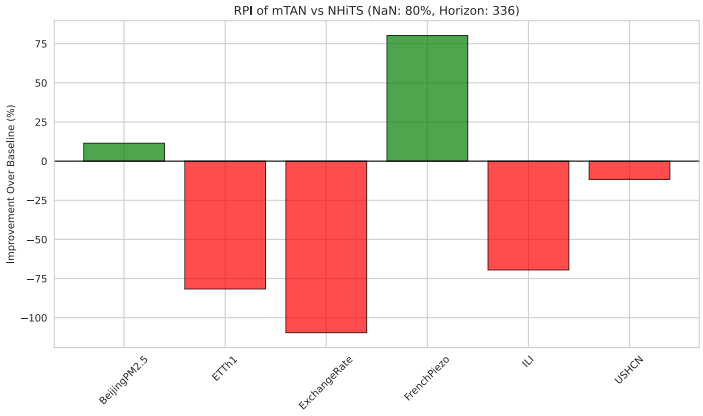
Figure A.82: Relative Performance Improvement of mTAN over NHiTS with 50% missing data.



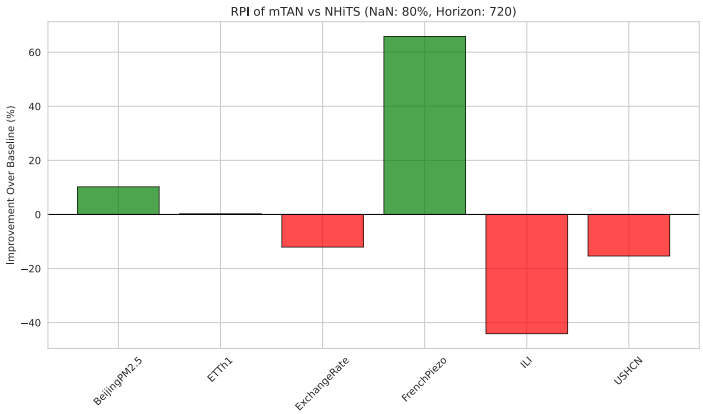
(a)



(b)

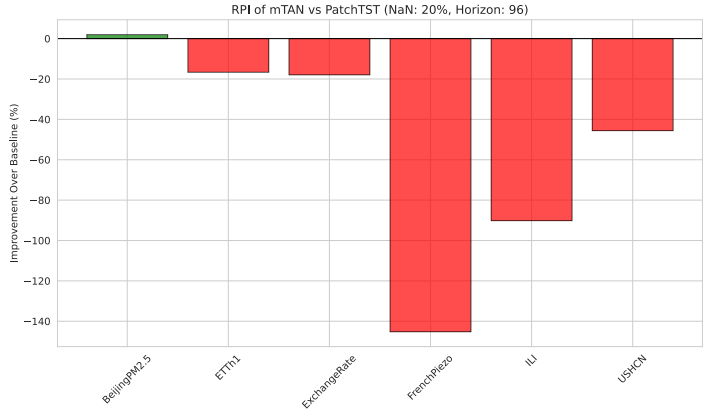


(c)

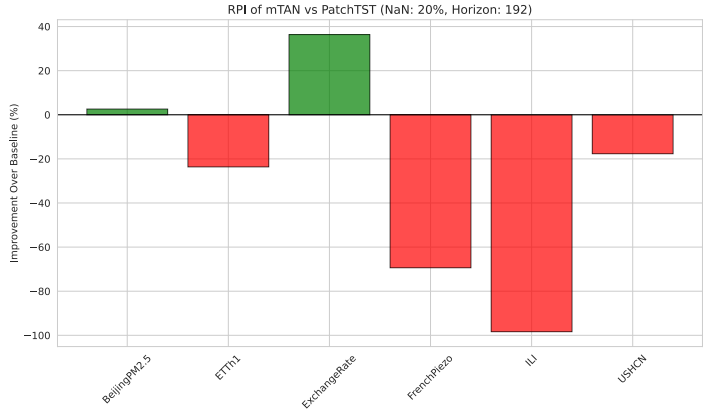


(d)

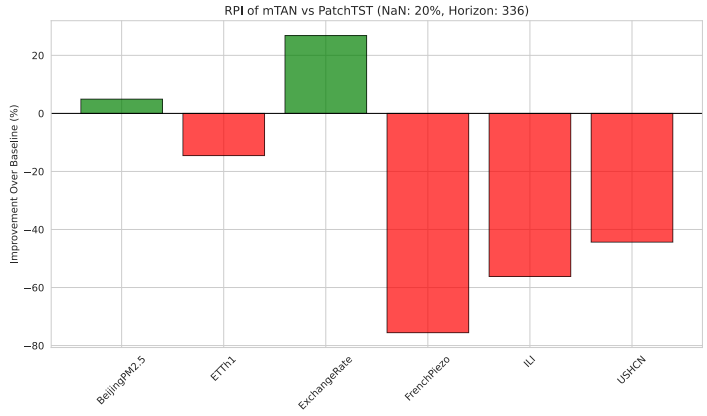
Figure A.83: Relative Performance Improvement of mTAN over NHiTS with 80% missing data.



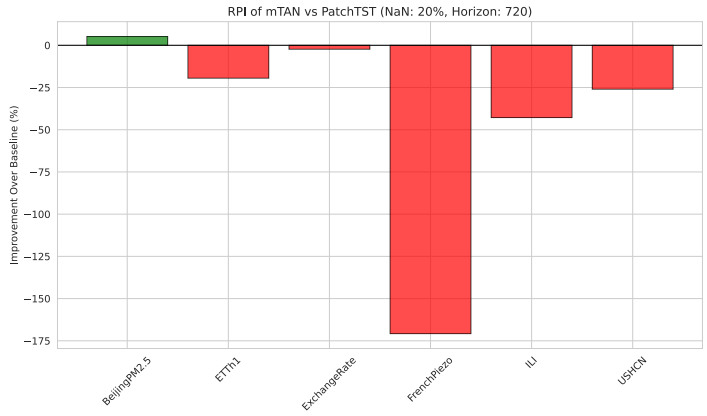
(a)



(b)

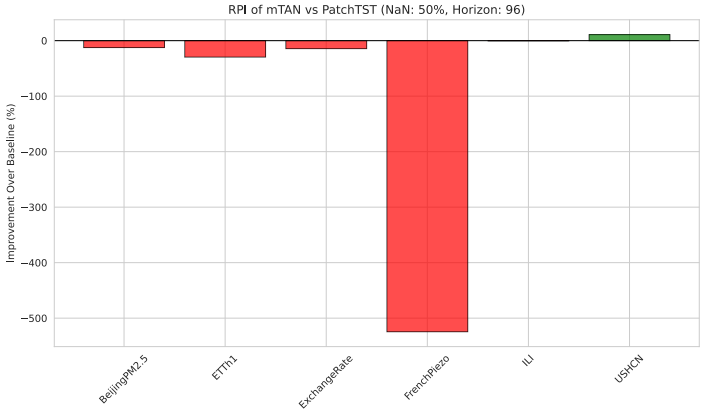


(c)

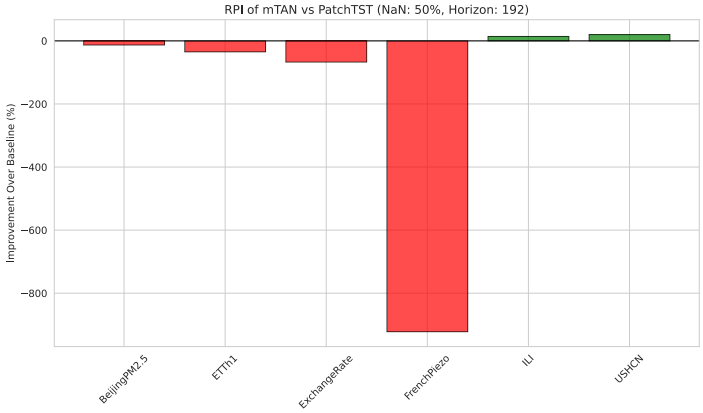


(d)

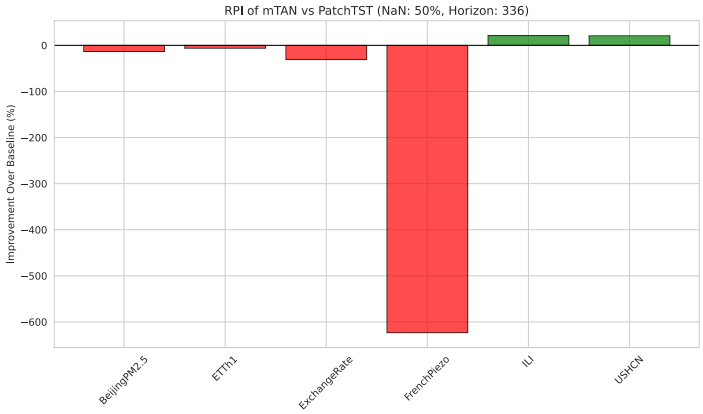
Figure A.84: Relative Performance Improvement of mTAN over PatchTST with 20% missing data.



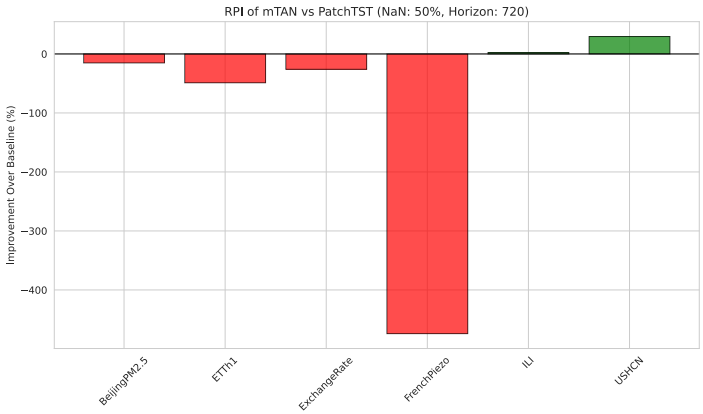
(a)



(b)

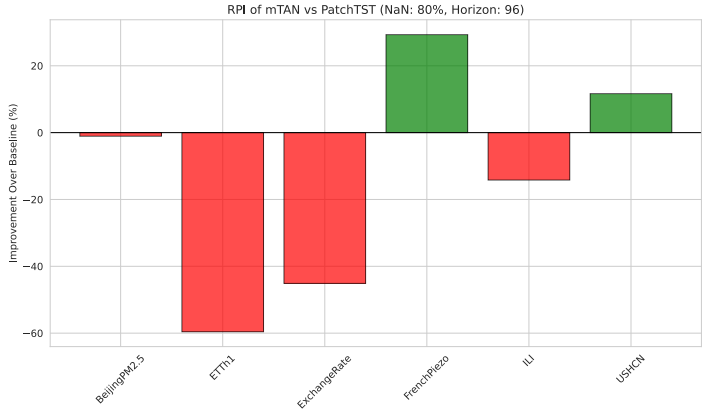


(c)

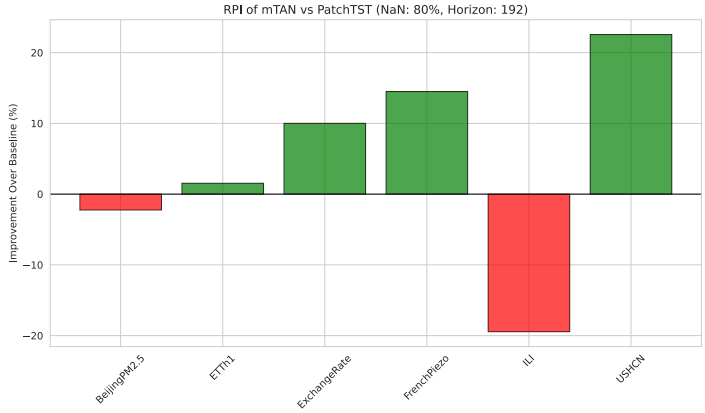


(d)

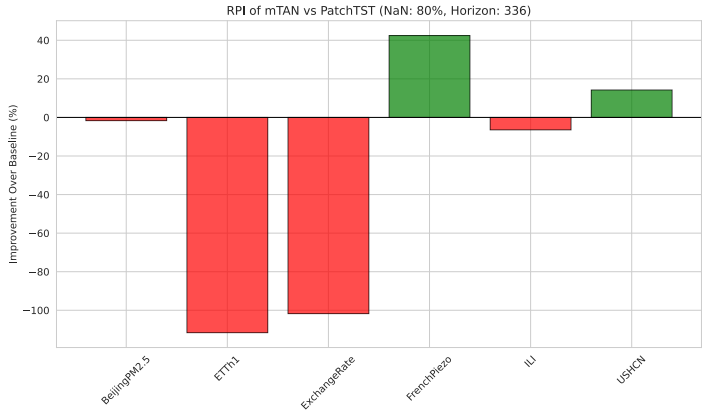
Figure A.85: Relative Performance Improvement of mTAN over PatchTST with 50% missing data.



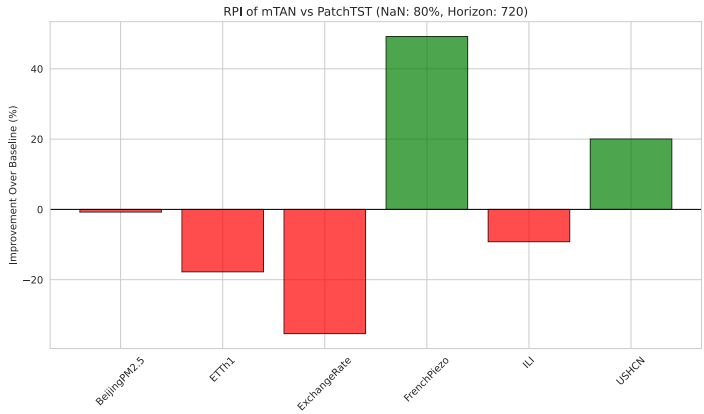
(a)



(b)

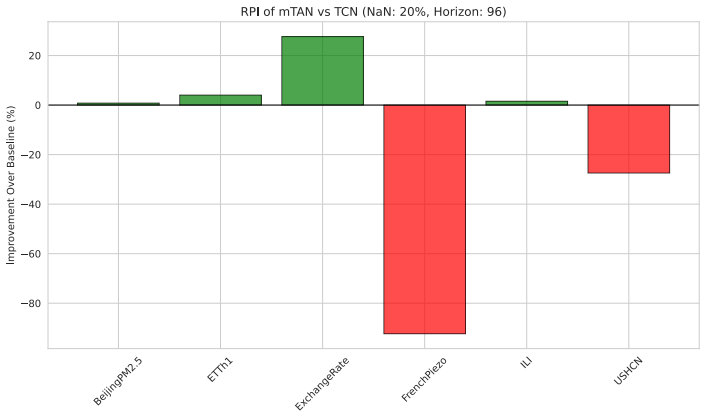


(c)

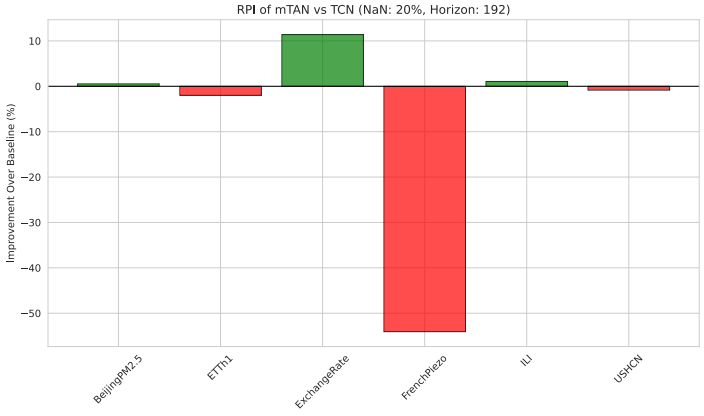


(d)

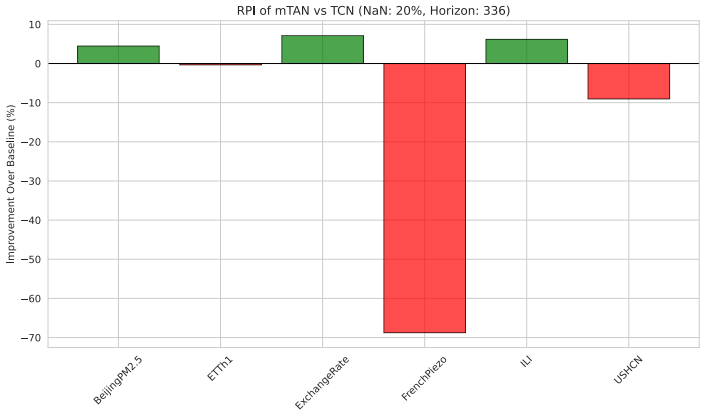
Figure A.86: Relative Performance Improvement of mTAN over PatchTST with 80% missing data.



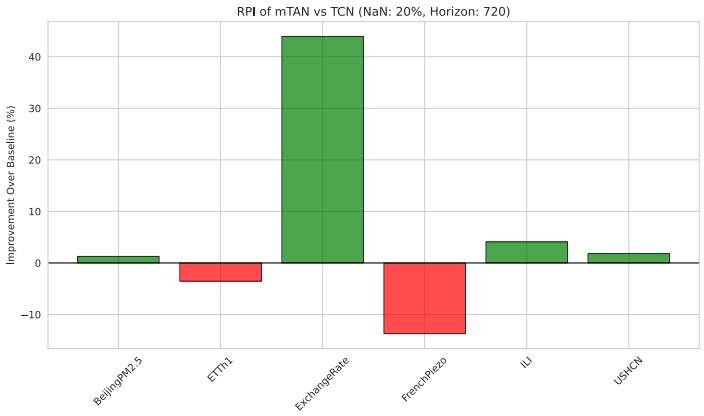
(a)



(b)

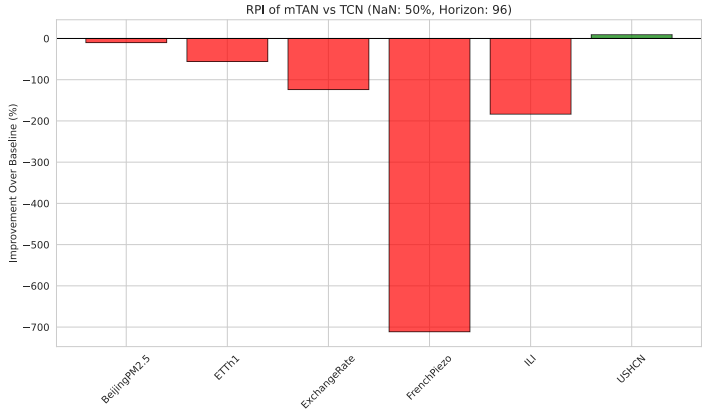


(c)

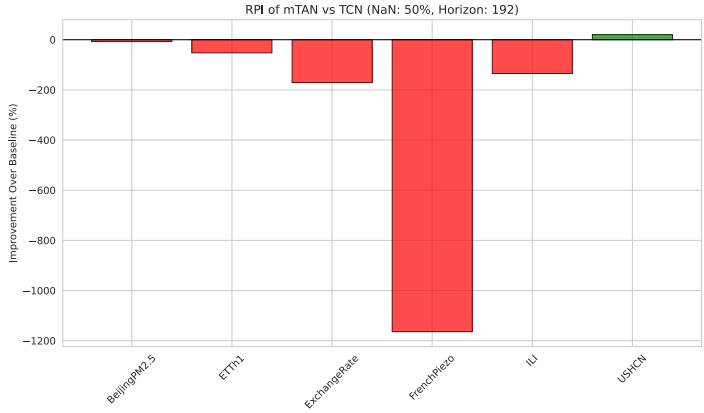


(d)

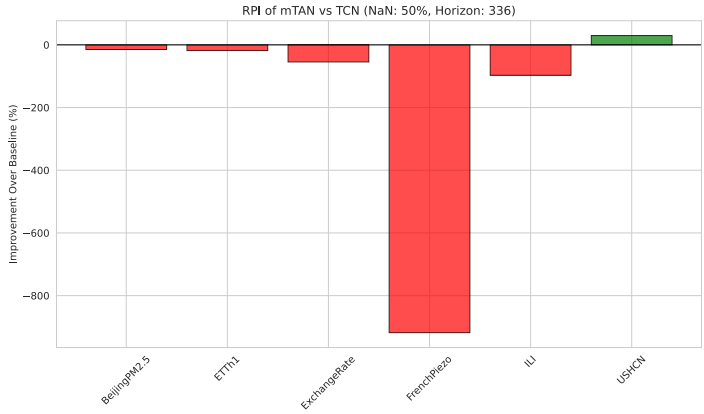
Figure A.87: Relative Performance Improvement of mTAN over TCN with 20% missing data.



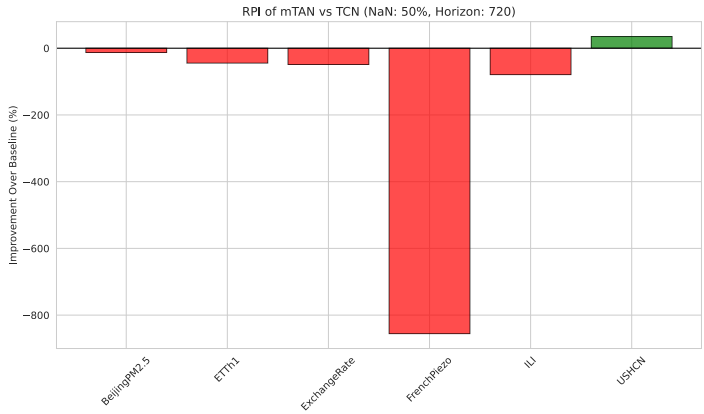
(a)



(b)

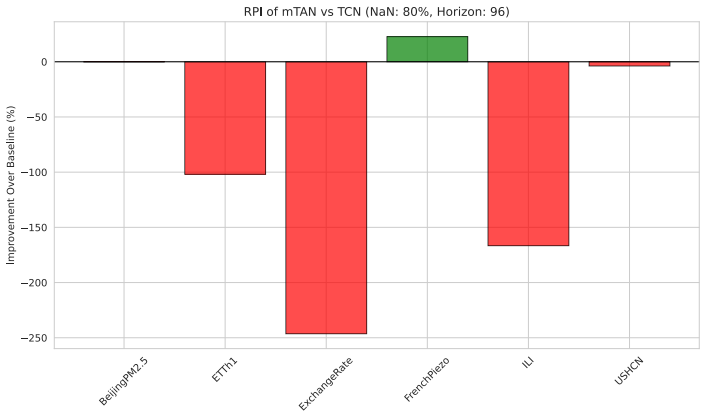


(c)

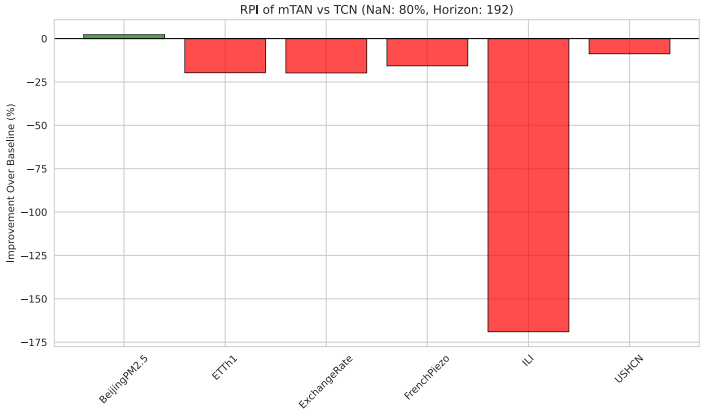


(d)

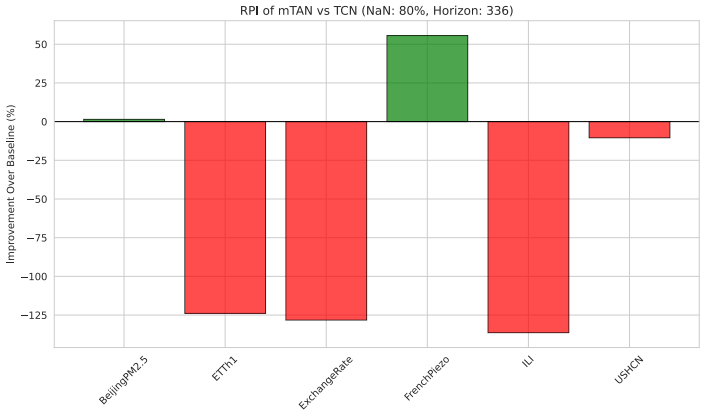
Figure A.88: Relative Performance Improvement of mTAN over TCN with 50% missing data.



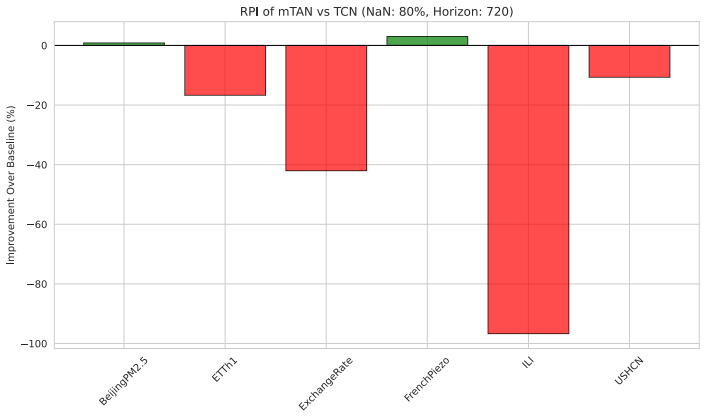
(a)



(b)



(c)



(d)

Figure A.89: Relative Performance Improvement of mTAN over TCN with 80% missing data.

A.4 RQ2.1

A.4.1 Imputation Models Hyperparameters Configurations

Listing A.1 reports the configuration found by the authors of the TSI-Bench / PyPOTS paper after running Microsoft's NNI to perform a grid search on each model's hyperparameters set to tailor the imputation performance to the input data [Du+24]. CSDI was not used because of performance issues while training, in particular an excessive usage of VRAM caused Out of Memory errors, so it was not possible to reliably train the model without running on CPU and requiring an excessive amount of time. The configurations found through this framework are too many to be reported in this document, as the configurations cannot be averaged over five runs as it was done for the metrics values. Therefore, it would be a total of 420 configurations to be reported.


```
{
"Beijing": {
  "SAITS": {
      "n_steps": 24,
      "n_features": 132,
      "epochs": 100,
      "patience": 10,
      "n_layers": 1,
      "d_model": 64,
      "d_ffn": 256,
      "n_heads": 4,
      "d_k": 256,
      "d_v": 64,
      "dropout": 0.1,
      "attn_dropout": 0.1,
      "lr": 0.00125059883141597
  },
  "BRITS": {
      "n_steps": 24,
      "n_features": 132,
      "patience": 10,
      "epochs": 100,
      "rnn_hidden_size": 512,
      "lr": 0.006242554068503864
  },
  "CSDI": {
      "n_steps": 24,
      "n_features": 132,
      "patience": 10,
      "epochs": 100,
      "n_layers": 6,
      "n_heads": 8,
      "n_channels": 32,
      "d_time_embedding": 256,
      "d_feature_embedding": 16,
      "d_diffusion_embedding": 32,
      "lr": 0.0036662098229766093
  },
  "FreTS": {
      "n_steps": 24,
      "n_features": 132,
      "epochs": 100,
      "patience": 10,
      "embed_size": 128,
      "hidden_size": 256,
      "channel_independence": false,
      "lr": 0.0013412816847317107
  },
  "USGAN": {
      "n_steps": 24,
      "n_features": 132,
      "patience": 10,
      "epochs": 100,
      "lr": 0.0005217674009036597,
      "rnn_hidden_size": 512,
      "dropout": 0.1
  },
  "DLinear": {
      "n_steps": 24,
      "n_features": 132,
      "epochs": 100,
      "patience": 10,
      "moving_avg_window_size": 5,
      "d_model": 256,
      "lr": 0.007545619490239286
  },
  "StemGNN": {
      "n_steps": 24,
      "n_features": 132,
      "epochs": 100,
      "patience": 10,
      "n_layers": 3,
      "n_stacks": 2,
      "d_model": 512,
      "dropout": 0,
      "lr": 0.002346769078880226
  },
  "TimesNet": {
      "n_steps": 24,
      "n_features": 132,
      "patience": 10,
      "epochs": 100,
      "n_layers": 2,
      "top_k": 1,
```

```
        "d_model": 1024,
        "d_ffn": 128,
        "n_kernels": 5,
        "dropout": 0.1,
        "lr": 0.000737615359931056
    },
  },
  "Electricity": {
    "SAITS": {
      "n_steps": 96,
      "n_features": 370,
      "epochs": 100,
      "patience": 10,
      "n_layers": 2,
      "d_model": 64,
      "d_ffn": 512,
      "n_heads": 8,
      "d_k": 256,
      "d_v": 128,
      "dropout": 0,
      "attn_dropout": 0.5,
      "lr": 0.0003804339175723239
    },
    "BRITS": {
        "n_steps": 96,
        "n_features": 370,
        "patience": 10,
        "epochs": 100,
        "rnn_hidden_size": 1024,
        "lr": 0.000648986719843512,
    },
    "CSDI": {
        "n_steps": 96,
        "n_features": 370,
        "patience": 10,
        "epochs": 100,
        "n_layers": 3,
        "n_heads": 4,
        "n_channels": 16,
        "d_time_embedding": 128,
        "d_feature_embedding": 8,
        "d_diffusion_embedding": 32,
        "lr": 0.0019998503932952497,
    },
    "FreTS": {
        "n_steps": 96,
        "n_features": 370,
        "epochs": 100,
        "patience": 10,
        "embed_size": 256,
        "hidden_size": 128,
        "channel_independence": false,
        "lr": 0.0006187379500510781,
    },
    "USGAN": {
        "n_steps": 96,
        "n_features": 370,
        "patience": 10,
        "epochs": 100,
        "lr": 0.00031344111157861616,
        "rnn_hidden_size": 512,
        "dropout": 0.3,
    },
    "DLinear": {
        "n_steps": 96,
        "n_features": 370,
        "epochs": 100,
        "patience": 10,
        "moving_avg_window_size": 5,
        "d_model": 1024,
        "lr": 0.001026937551416374,
    },
    "StemGNN": {
        "n_steps": 96,
        "n_features": 370,
        "epochs": 100,
        "patience": 10,
        "n_layers": 2,
        "n_stacks": 2,
        "d_model": 1024,
        "dropout": 0,
        "lr": 0.00019510837269346798,
    },
    "TimesNet": {
        "n_steps": 96,
        "n_features": 370,
        "patience": 10,
        "epochs": 100,
        "n_layers": 2,
        "top_k": 4,
        "d_model": 1024,
        "d_ffn": 128,
        "n_kernels": 4,
        "dropout": 0,
        "lr": 0.00013095401869683552,
    },
  },
  "ETTh1": {
    "SAITS": {
      "n_steps": 48,
      "n_features": 7,
      "epochs": 100,
      "patience": 10,
      "n_layers": 3,
      "d_model": 64,
      "d_ffn": 512,
      "n_heads": 8,
      "d_k": 256,
      "d_v": 128,
      "dropout": 0,
      "attn_dropout": 0.1,
      "lr": 0.00013619330183926336
    },
    "BRITS": {
```

```
        "n_steps": 48,
        "n_features": 7,
        "patience": 10,
        "epochs": 100,
        "rnn_hidden_size": 512,
        "lr": 0.005364156231811175,
    },
    "CSDI": {
        "n_steps": 48,
        "n_features": 7,
        "patience": 10,
        "epochs": 100,
        "n_layers": 4,
        "n_heads": 4,
        "n_channels": 128,
        "d_time_embedding": 128,
        "d_feature_embedding": 16,
        "d_diffusion_embedding": 128,
        "lr": 0.0012659791476320047,
    },
    "FreTS": {
        "n_steps": 48,
        "n_features": 7,
        "epochs": 100,
        "patience": 10,
        "embed_size": 128,
        "hidden_size": 64,
        "channel_independence": true,
        "lr": 0.007511355522399692,
    },
    "USGAN": {
        "n_steps": 48,
        "n_features": 7,
        "patience": 10,
        "epochs": 100,
        "lr": 0.009769718248164798,
        "rnn_hidden_size": 512,
        "dropout": 0.4,
    },
    "DLinear": {
        "n_steps": 48,
        "n_features": 7,
        "epochs": 100,
        "patience": 10,
        "moving_avg_window_size": 13,
        "d_model": 64,
        "lr": 0.003881222990953312,
    },
    "StemGNN": {
        "n_steps": 48,
        "n_features": 7,
        "epochs": 100,
        "patience": 10,
        "n_layers": 2,
        "n_stacks": 2,
        "d_model": 1024,
        "dropout": 0.1,
        "lr": 0.0025092583072368864,
    },
    "TimesNet": {
        "n_steps": 48,
        "n_features": 7,
        "patience": 10,
        "epochs": 100,
        "n_layers": 1,
        "top_k": 1,
        "d_model": 128,
        "d_ffn": 256,
        "n_kernels": 4,
        "dropout": 0.1,
        "lr": 0.0004496195292510804,
    }
  }
}
```


Listing A.1: Configuration selected from the authors of the TSI-Bench paper as the result of a grid search over the models hyperparameters [Du+24].

A.4.2 Raw Performance Metrics

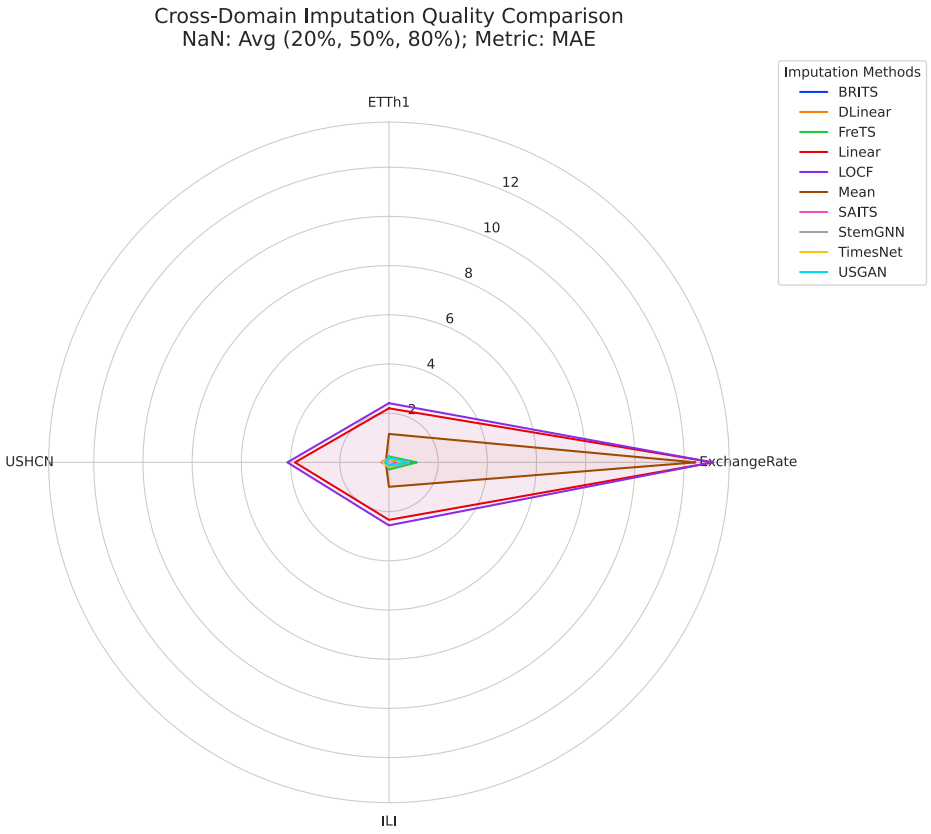
Table Table A.4 reports the mean values for MAE and MSE across all datasets, horizons, and sparsity levels (20%, 50%, 80%) averaged across 5 runs.

			Imputation Method																	
			Metric																	
			BRITS		DLinear		FreTS		Linear		LOCF		Mean		SAITS		StemGNN		TimesNet		USGAN	
			MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE
NaN Pattern Dataset	ETTh1	20%	0.100	0.148	0.138	0.170	0.082	0.113	0.574	0.482	0.621	0.501	0.289	0.339	0.070	0.095	0.134	0.156	0.103	0.149	0.138	0.189
		50%	0.246	0.322	0.177	0.213	0.328	0.385	1.580	1.368	1.756	1.436	0.845	0.990	0.290	0.336	0.267	0.265	0.259	0.350	0.260	0.324
		80%	0.155	0.247	0.185	0.404	0.314	0.668	4.443	3.815	4.842	3.959	2.335	2.709	0.215	0.388	0.168	0.261	0.162	0.241	0.202	0.309
	ExchangeRate	20%	0.609	0.438	0.458	0.330	0.639	0.350	3.312	1.291	3.306	1.290	3.126	1.290	0.211	0.189	0.523	0.375	0.454	0.350	0.612	0.451
		50%	1.335	0.956	0.312	0.270	1.931	1.035	9.579	3.733	9.548	3.726	8.982	3.722	0.362	0.334	1.403	0.985	0.628	0.498	0.695	0.517
		80%	0.823	0.655	0.244	0.209	0.853	0.827	26.650	10.343	26.622	10.340	25.275	10.364	0.509	0.508	0.597	0.422	0.479	0.373	0.926	0.787
	ILI	20%	0.097	0.138	0.059	0.088	0.271	0.266	0.635	0.494	0.665	0.500	0.257	0.344	0.043	0.073	0.202	0.175	0.019	0.047	0.053	0.084
		50%	0.243	0.292	0.185	0.216	0.234	0.269	1.796	1.445	1.939	1.472	0.752	1.013	0.171	0.235	0.231	0.210	0.248	0.307	0.215	0.282
		80%	0.162	0.330	0.163	0.329	0.366	0.614	4.582	3.760	5.077	3.841	1.971	2.746	0.158	0.231	0.228	0.355	0.330	0.580	0.137	0.232
	USHCN	20%	0.154	0.045	0.194	0.096	0.180	0.082	1.201	0.508	1.357	0.521	0.040	0.154	0.257	0.170	0.180	0.081	0.240	0.163	0.158	0.086
		50%	0.202	0.067	0.253	0.102	0.205	0.071	2.996	1.284	3.326	1.311	0.101	0.389	0.378	0.278	0.195	0.069	0.325	0.238	0.179	0.074
		80%	0.155	0.053	0.181	0.071	0.145	0.054	7.267	3.131	7.705	3.174	0.247	0.953	0.302	0.503	0.192	0.152	0.230	0.190	0.135	0.059

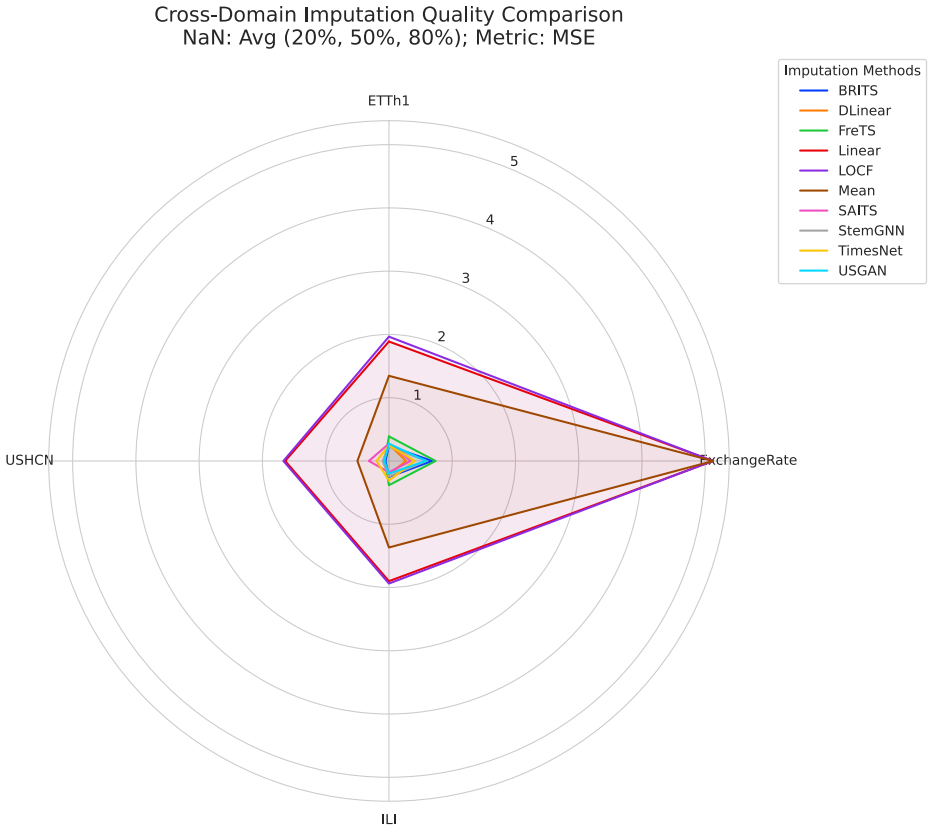
Table A.4: Raw MAE and MSE values gathered for RQ2.1 averaged across 5 runs.

A.4.3 Cross-Domain Radar Analysis

Radar plots illustrate model performance across the four primary domains (ETTh1, ExchangeRate, ILI, USHCN). First of all, plots where the metrics were averaged over missing data percentages are shown; subsequently, each plot represents a specific NaN% and metric.

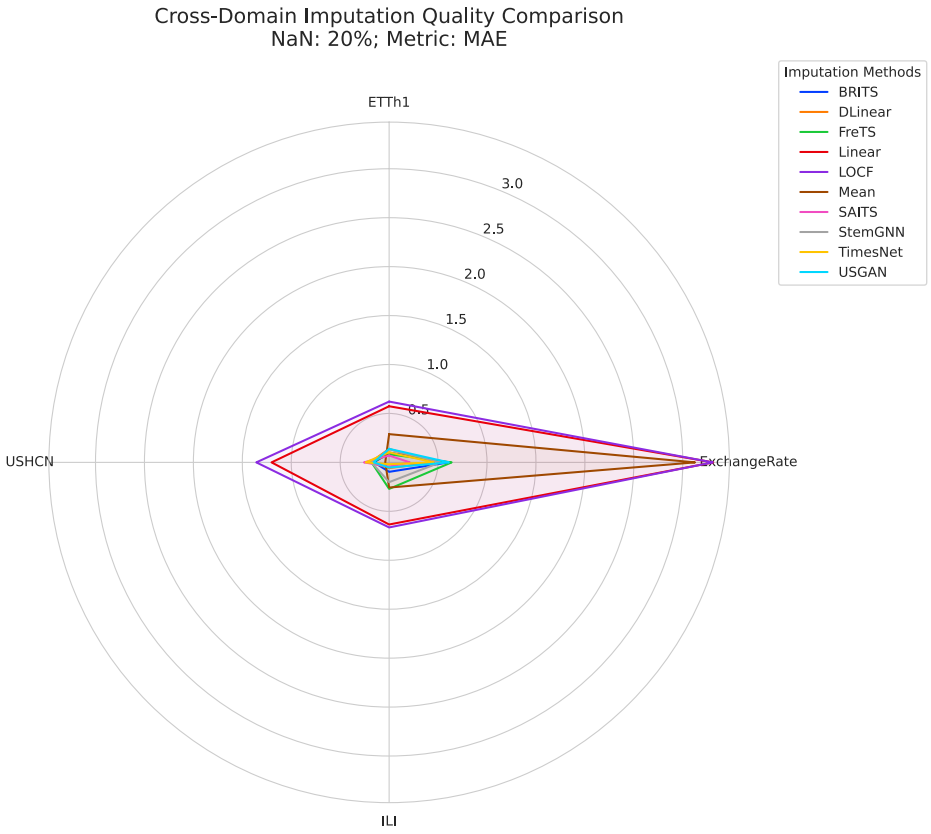


(a)

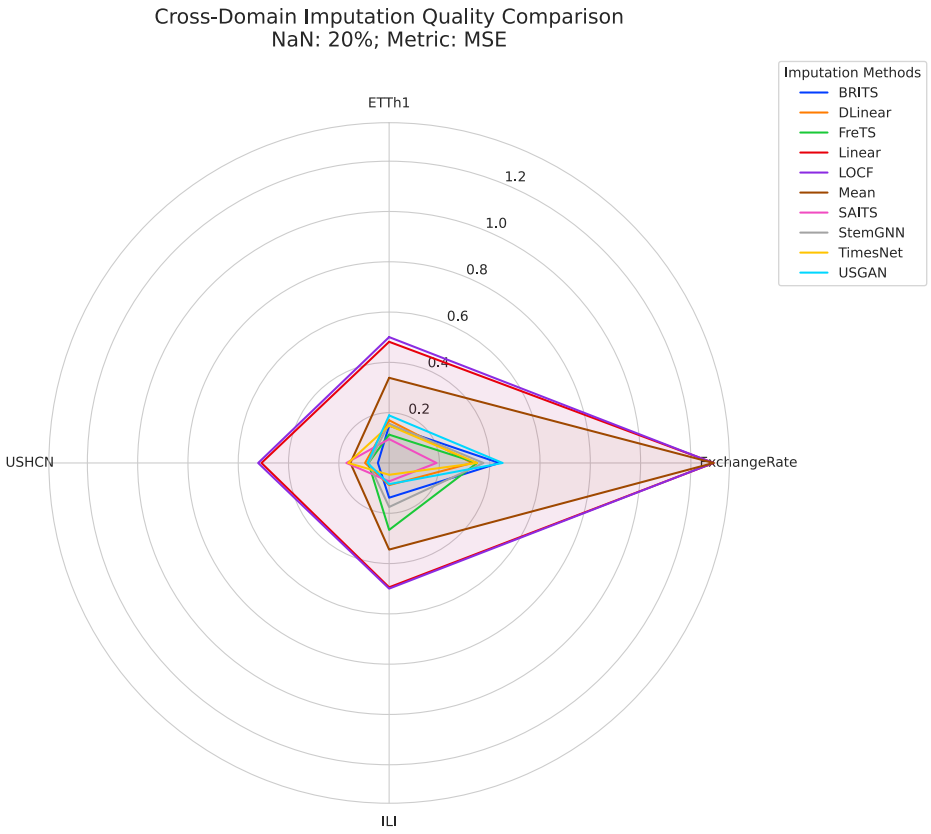


(b)

Figure A.90: Radar plot comparing imputation performance on all datasets across all methods and missing data percentages.

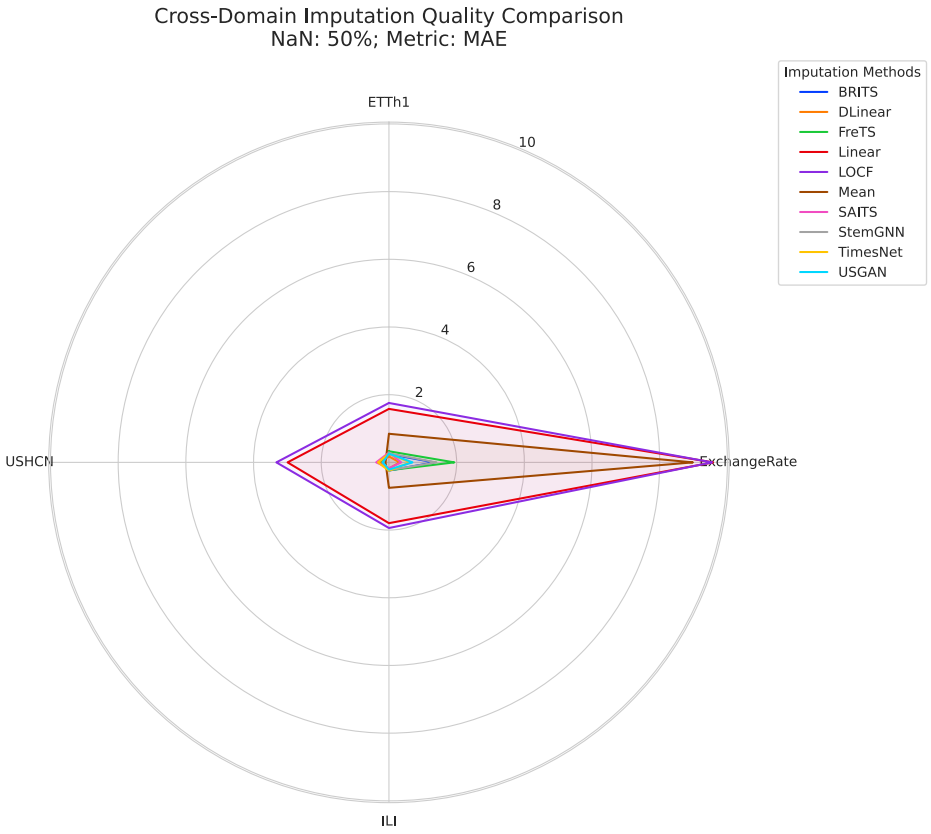


(a)

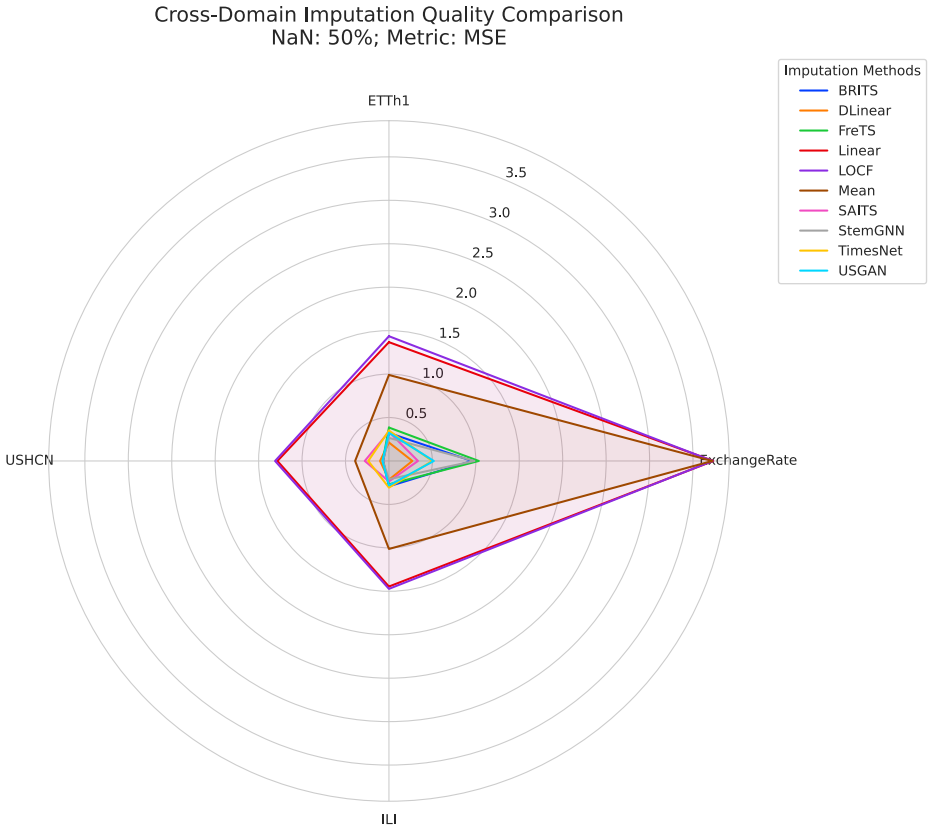


(b)

Figure A.91: Radar plot comparing imputation performance on all datasets across all methods with 20% missing data.

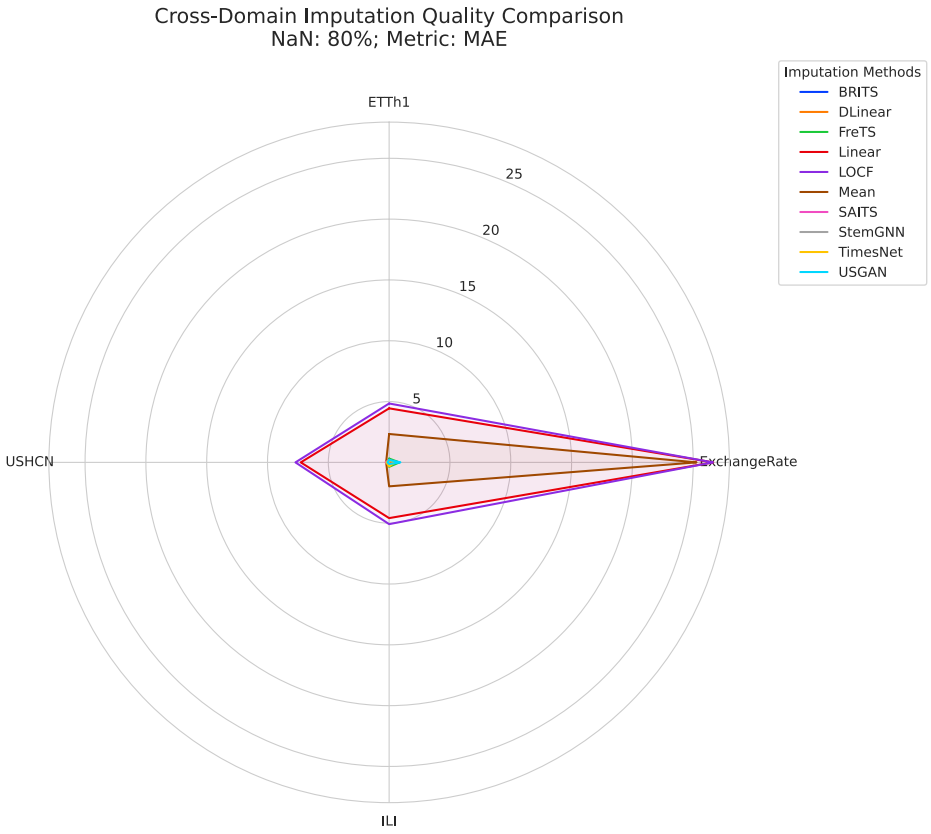


(a)

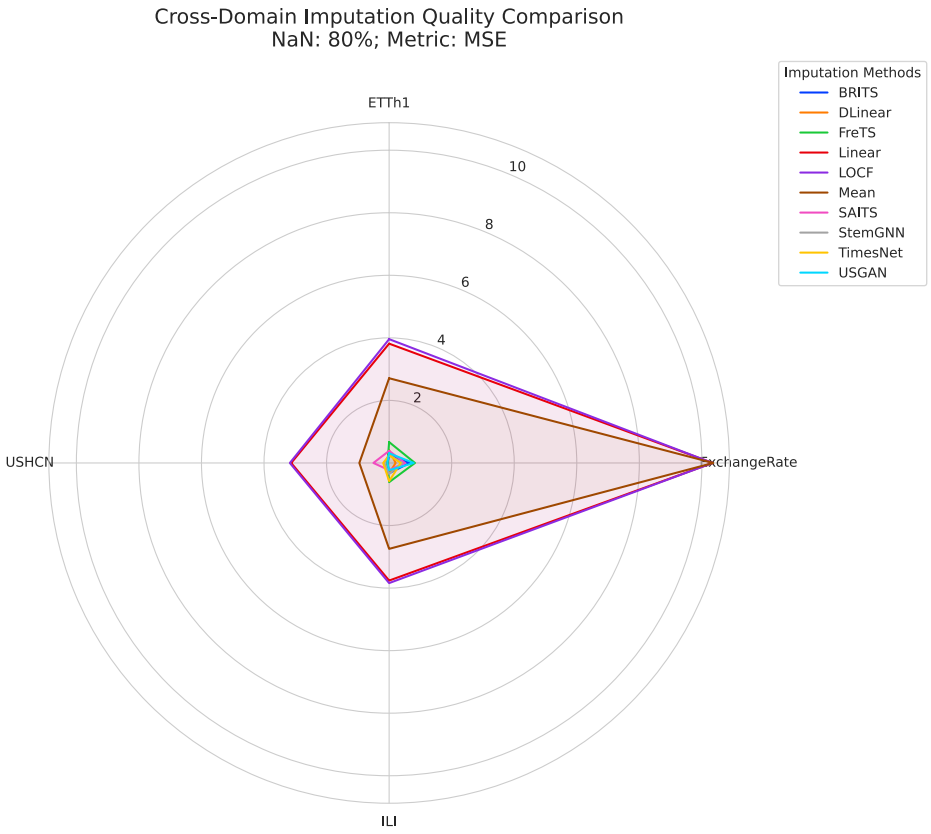


(b)

Figure A.92: Radar plot comparing imputation performance on all datasets across all methods with 50% missing data.

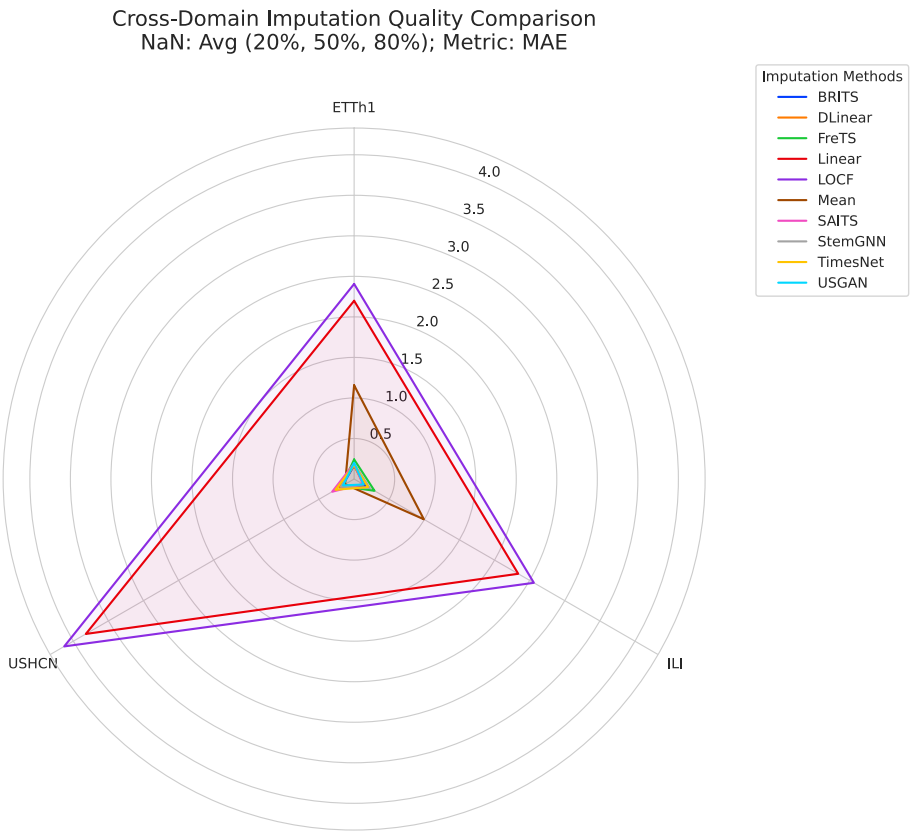


(a)

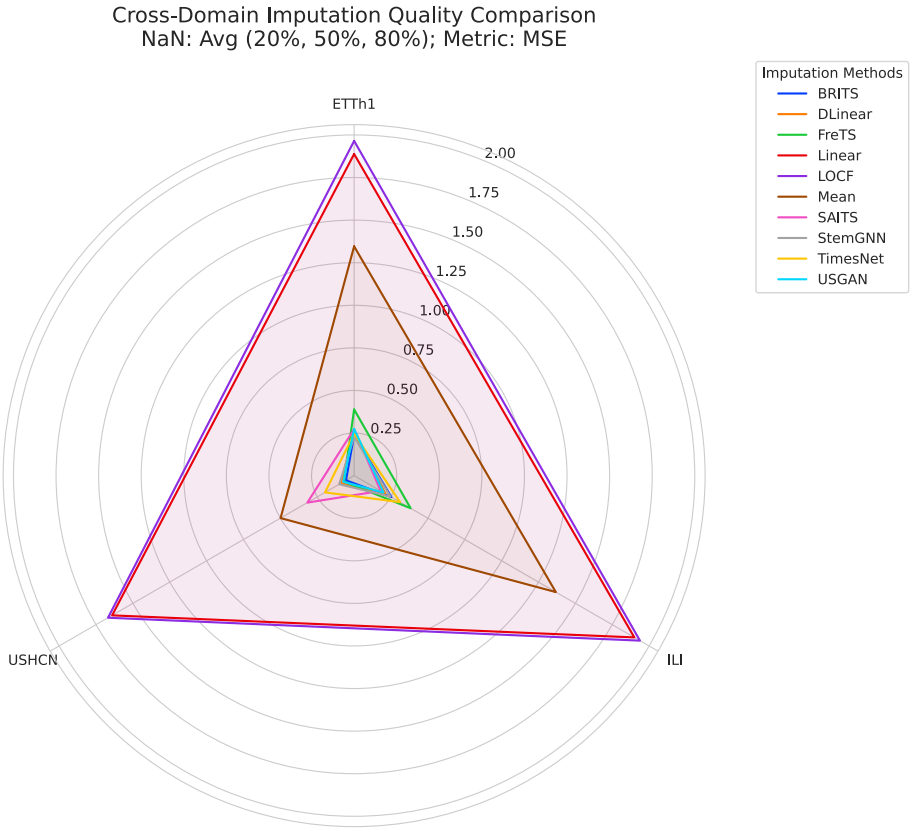


(b)

Figure A.93: Radar plot comparing imputation performance on all datasets across all methods with 80% missing data.

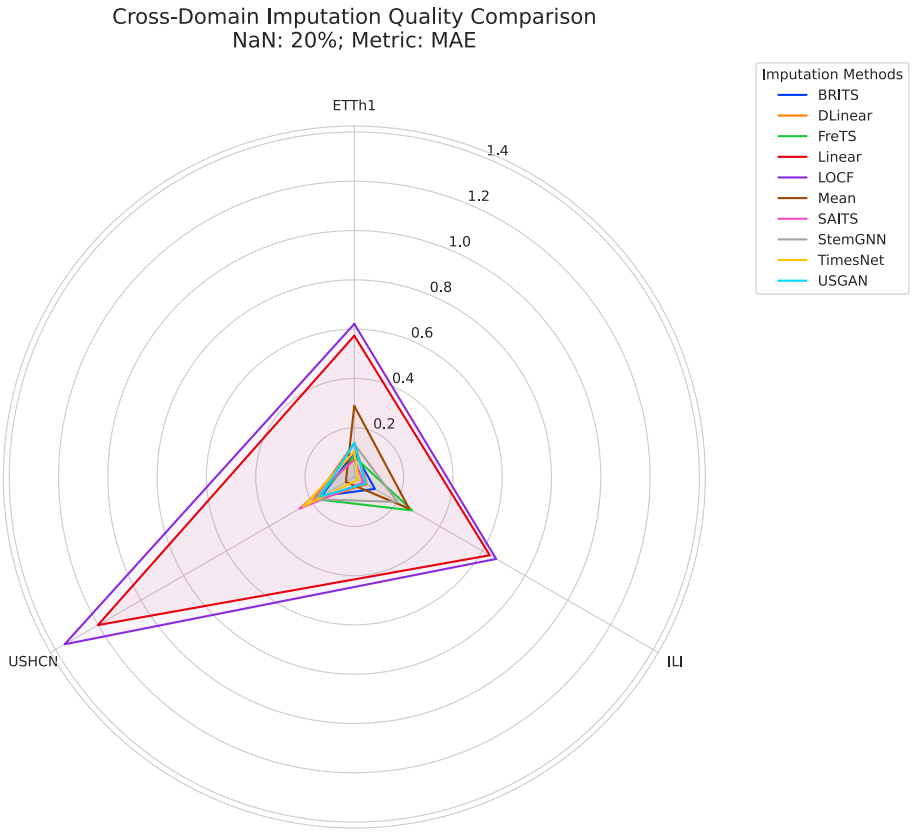


(a)

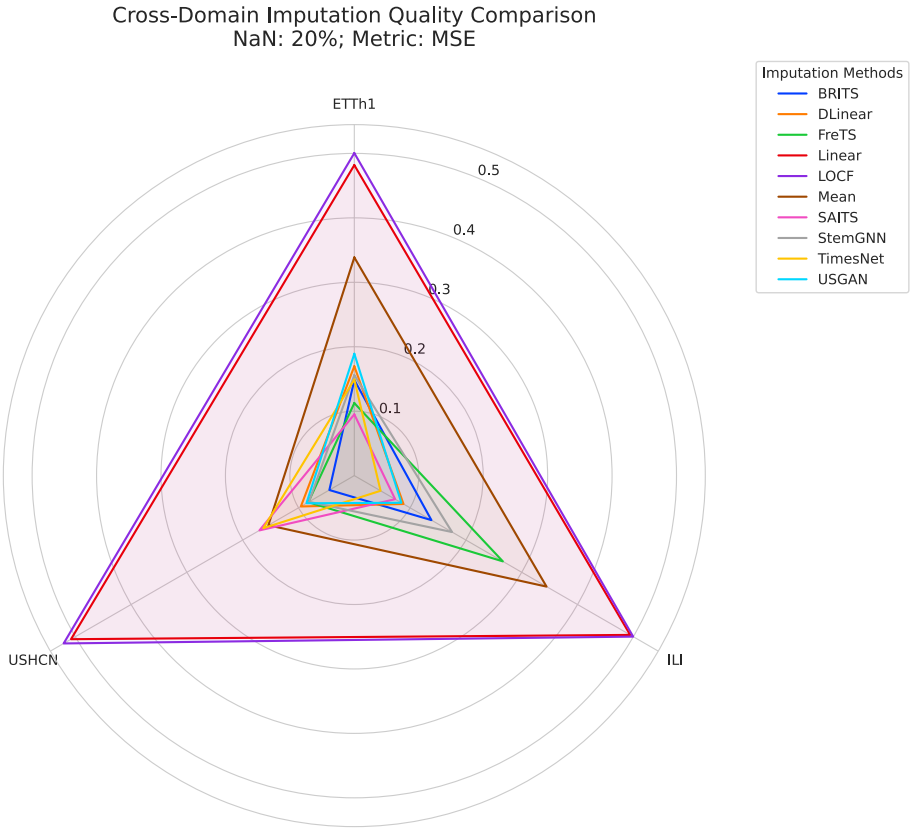


(b)

Figure A.94: Radar plot comparing imputation performance on all datasets **except** ExchangeRate across all methods and missing data percentages.

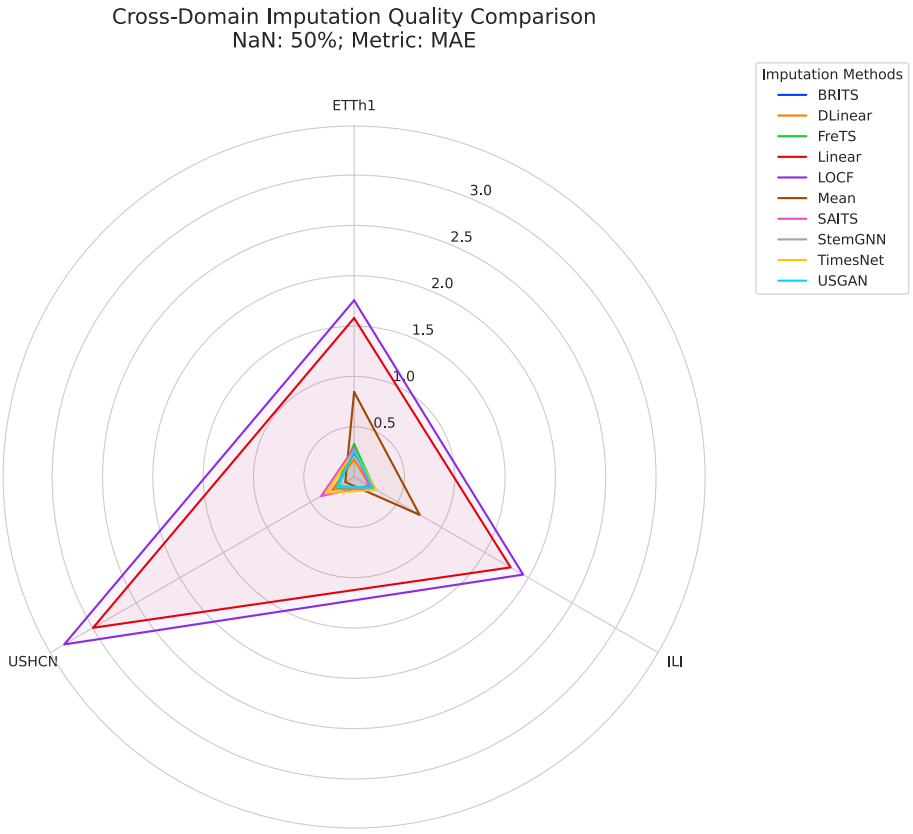


(a)

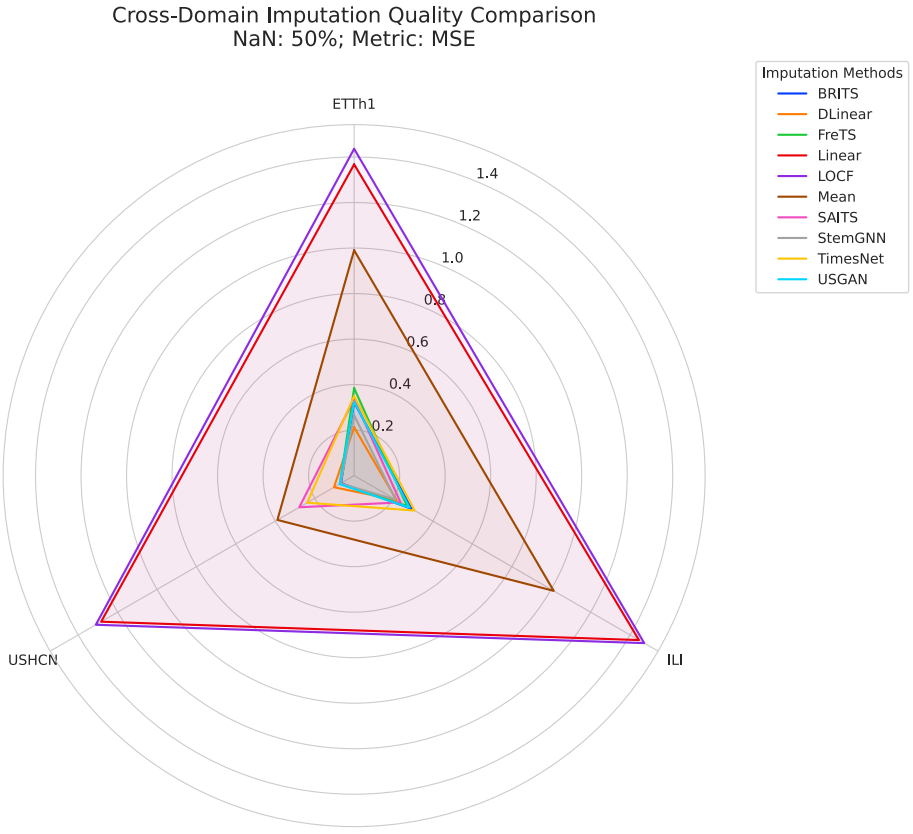


(b)

Figure A.95: Radar plot comparing imputation performance on all datasets **except** ExchangeRate across all methods with 20% missing data.

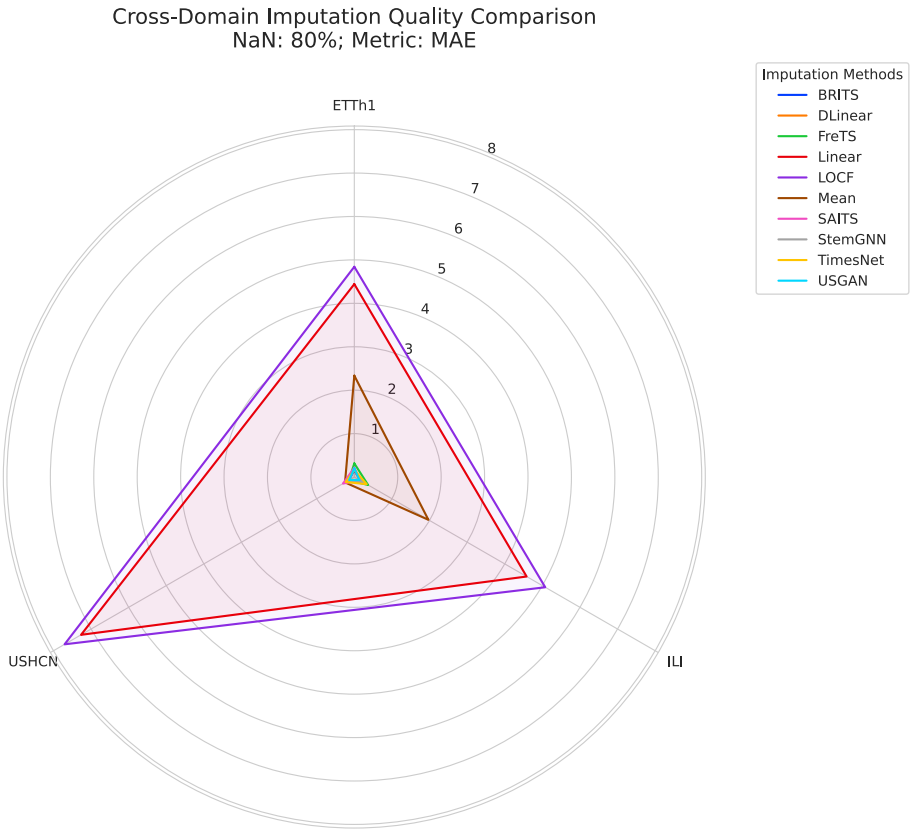


(a)

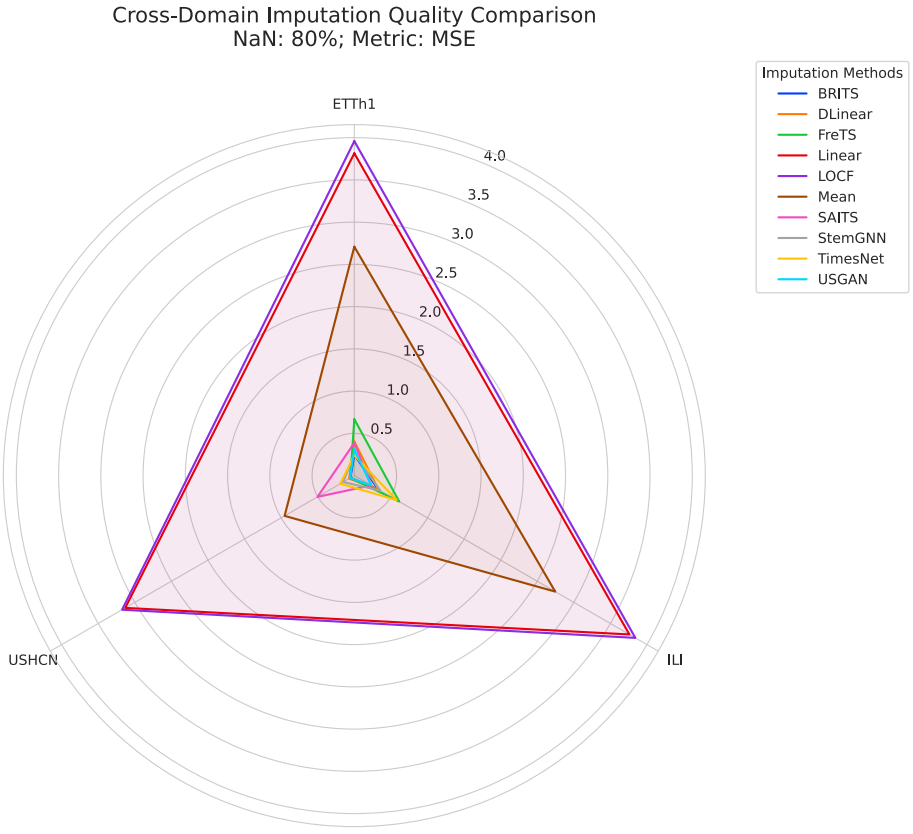


(b)

Figure A.96: Radar plot comparing imputation performance on all datasets **except** ExchangeRate across all methods with 50% missing data.

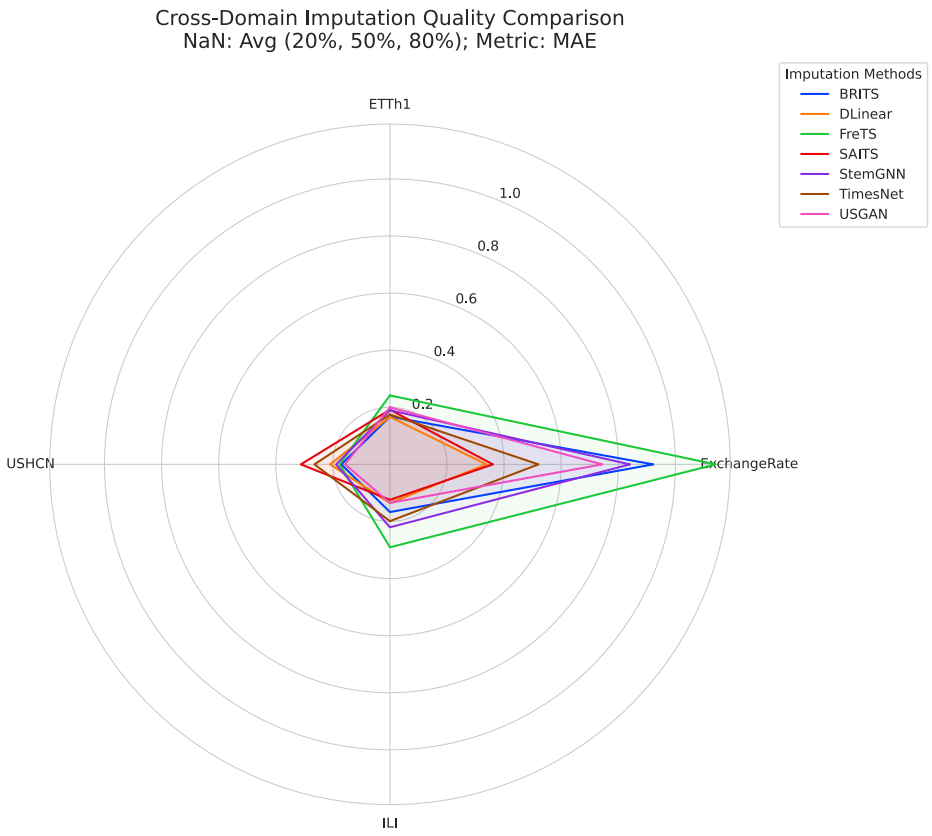


(a)

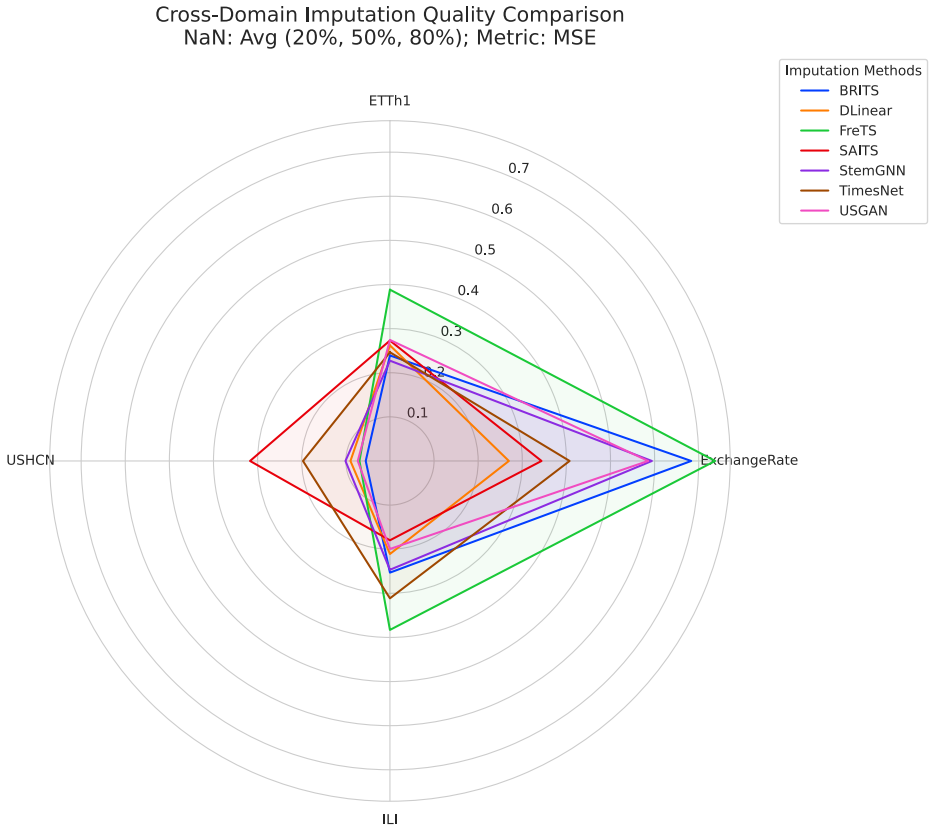


(b)

Figure A.97: Radar plot comparing imputation performance on all datasets **except** ExchangeRate across all methods with 80% missing data.

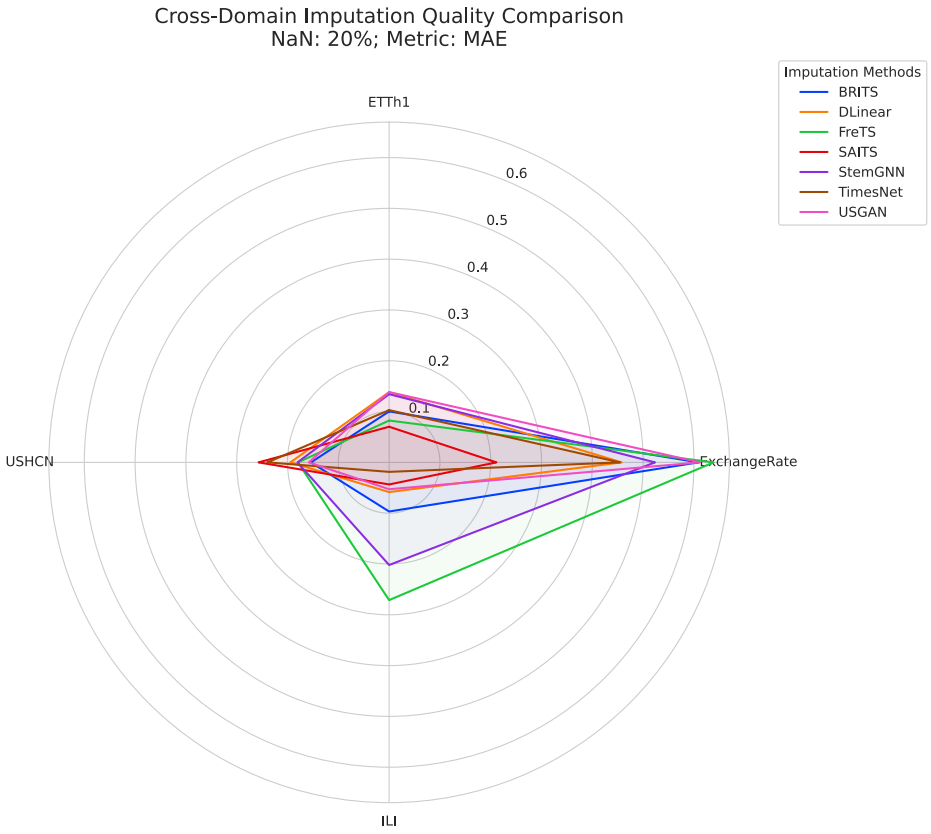


(a)

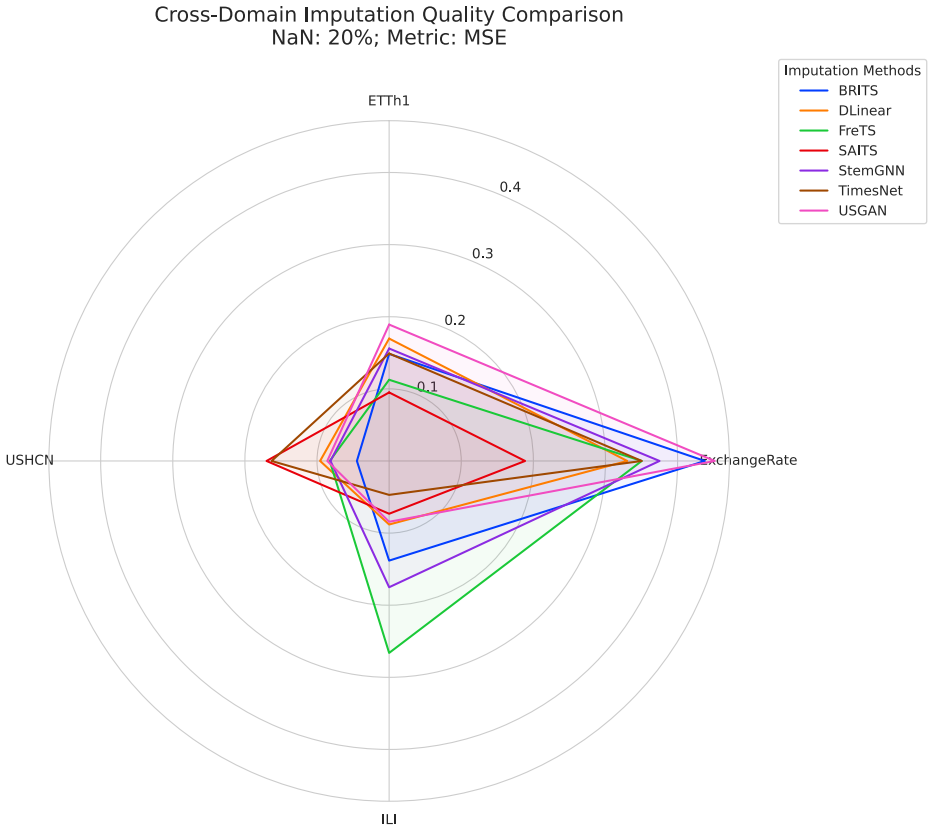


(b)

Figure A.98: Radar plot comparing imputation performance on all datasets across complex imputation methods and all missing data percentages.

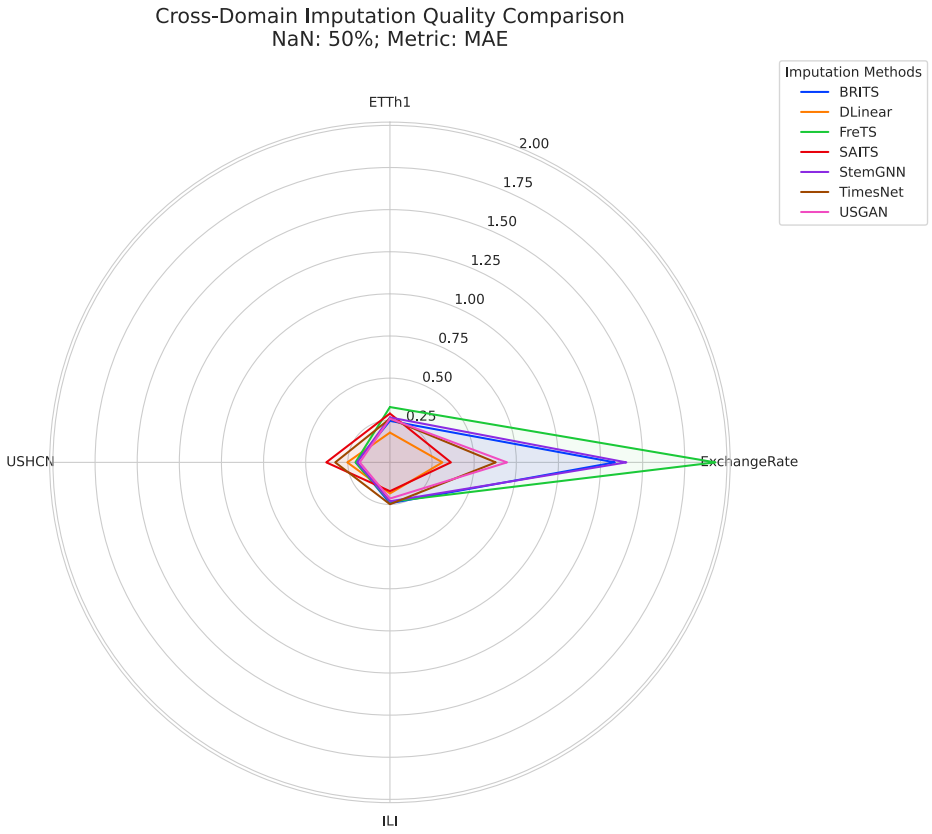


(a)

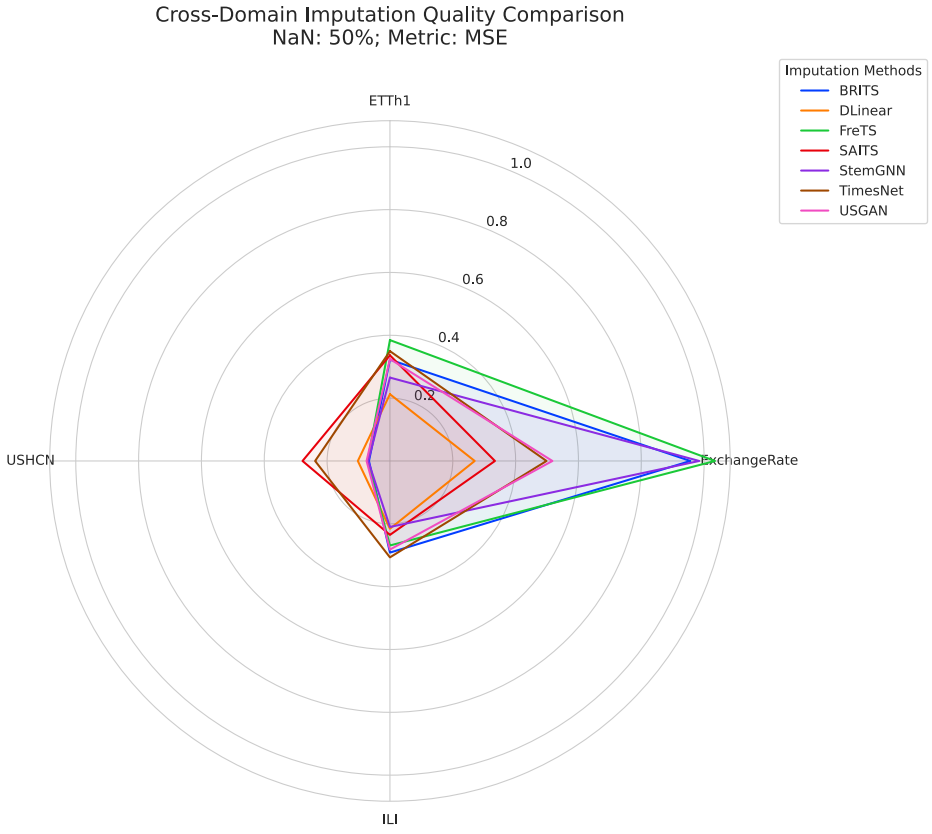


(b)

Figure A.99: Radar plot comparing imputation performance on all datasets across complex imputation methods with 20% missing data.

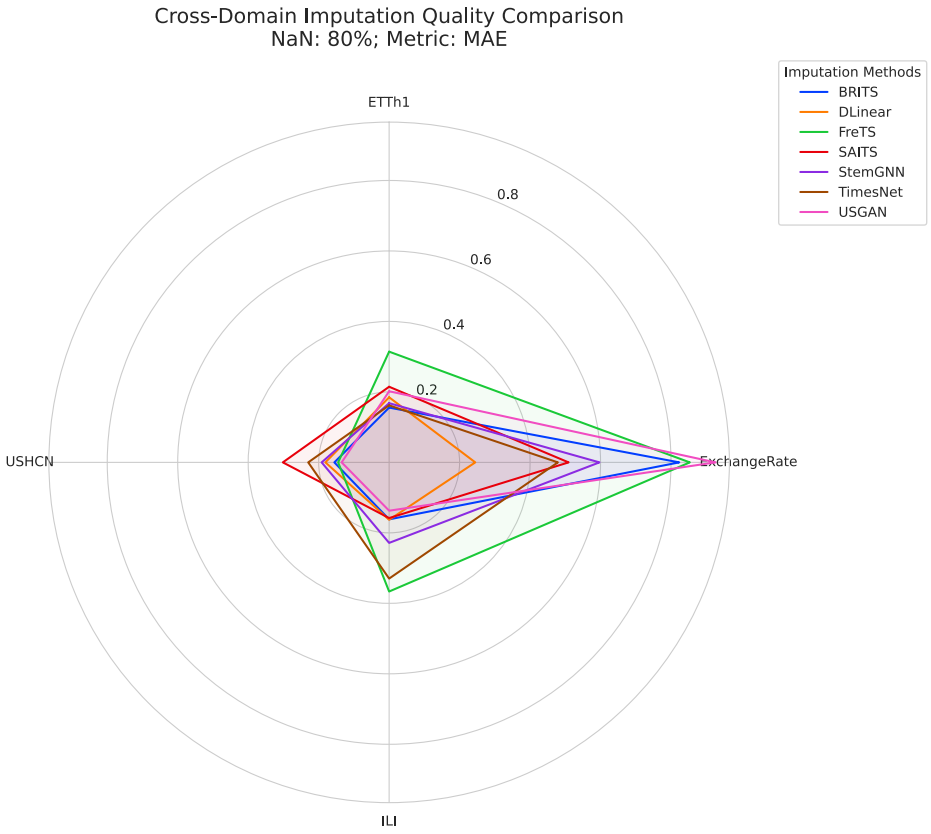


(a)

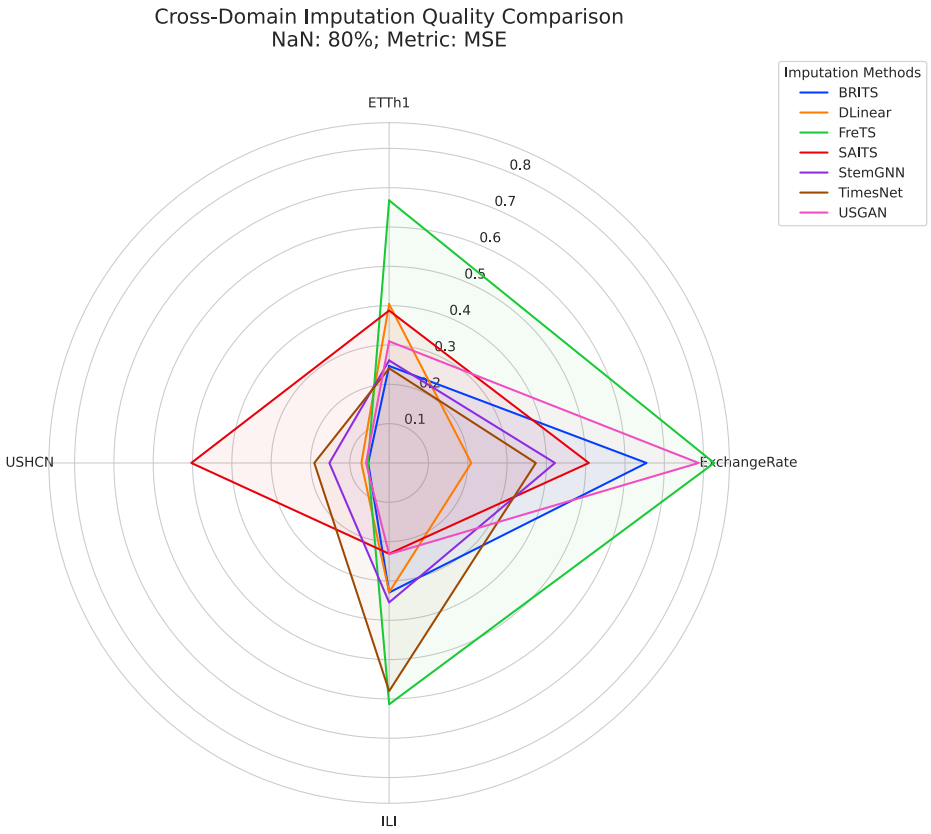


(b)

Figure A.100: Radar plot comparing imputation performance on all datasets across complex imputation methods with 50% missing data.

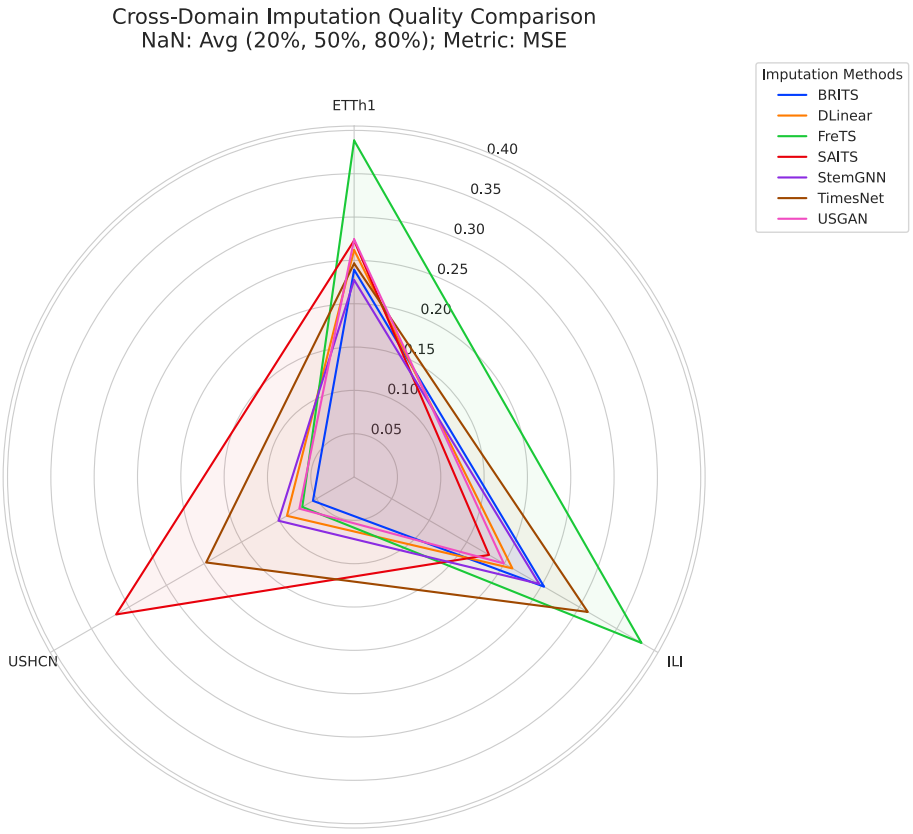


(a)

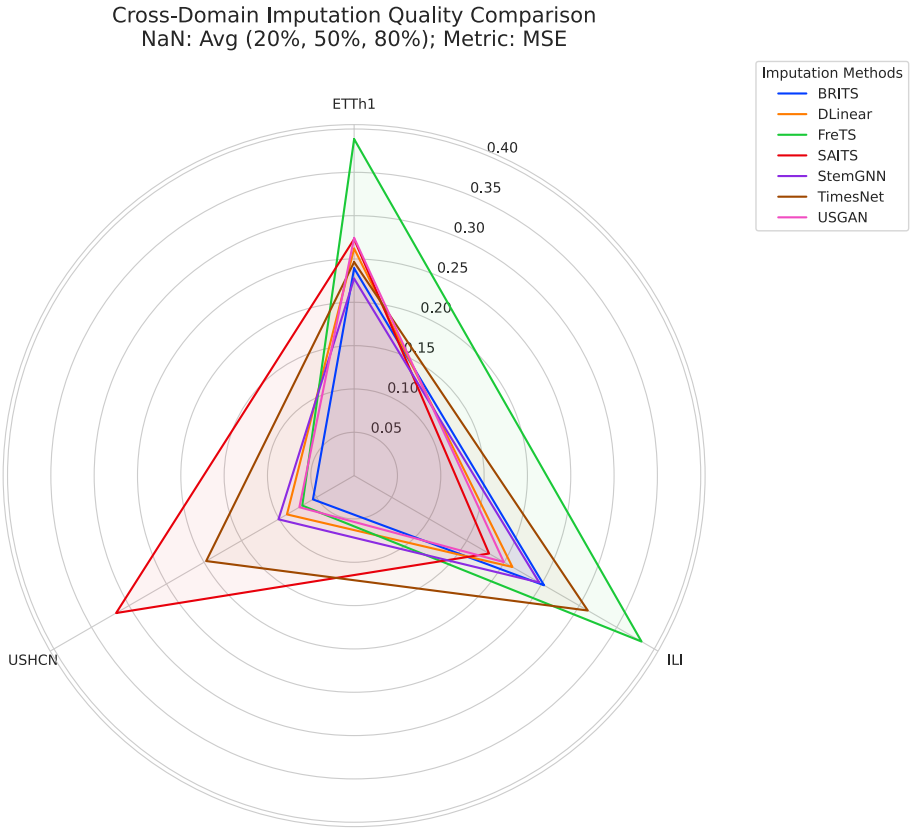


(b)

Figure A.101: Radar plot comparing imputation performance on all datasets across complex imputation methods with 80% missing data.

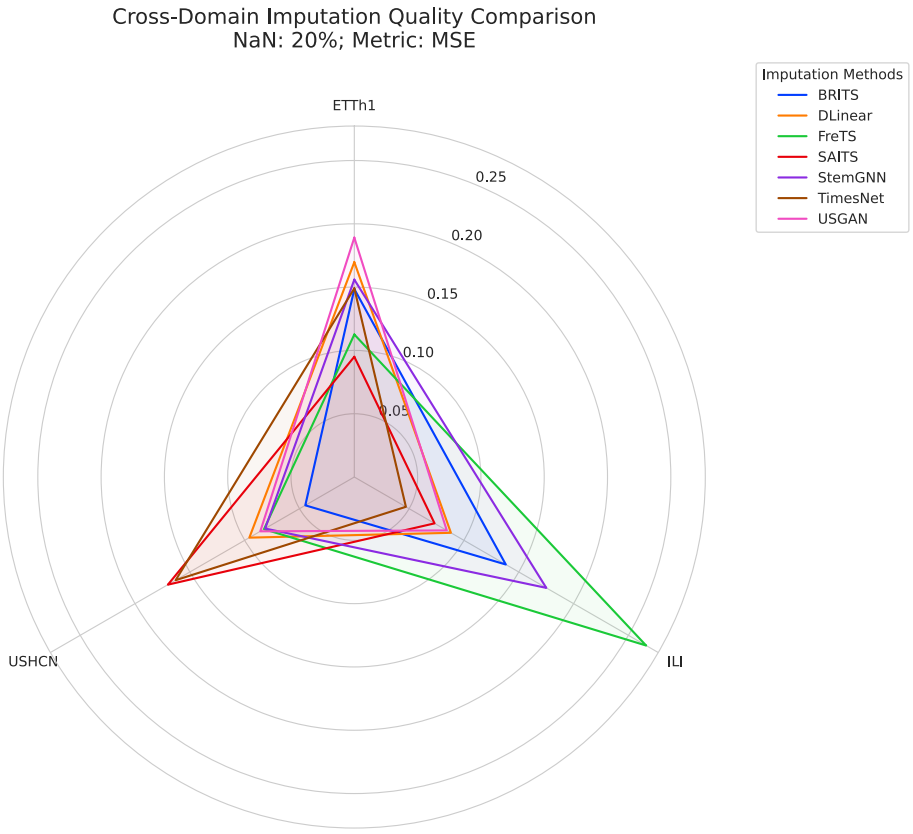


(a)

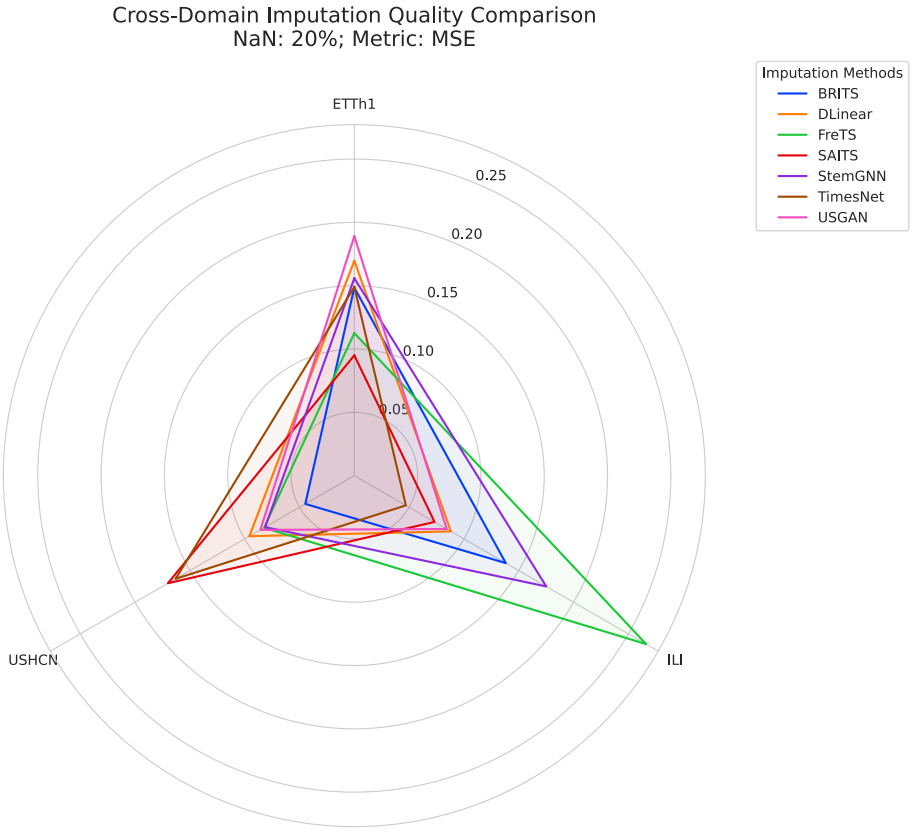


(b)

Figure A.102: Radar plot comparing imputation performance on all datasets **except** ExchangeRate across complex imputation methods and all missing data percentages.

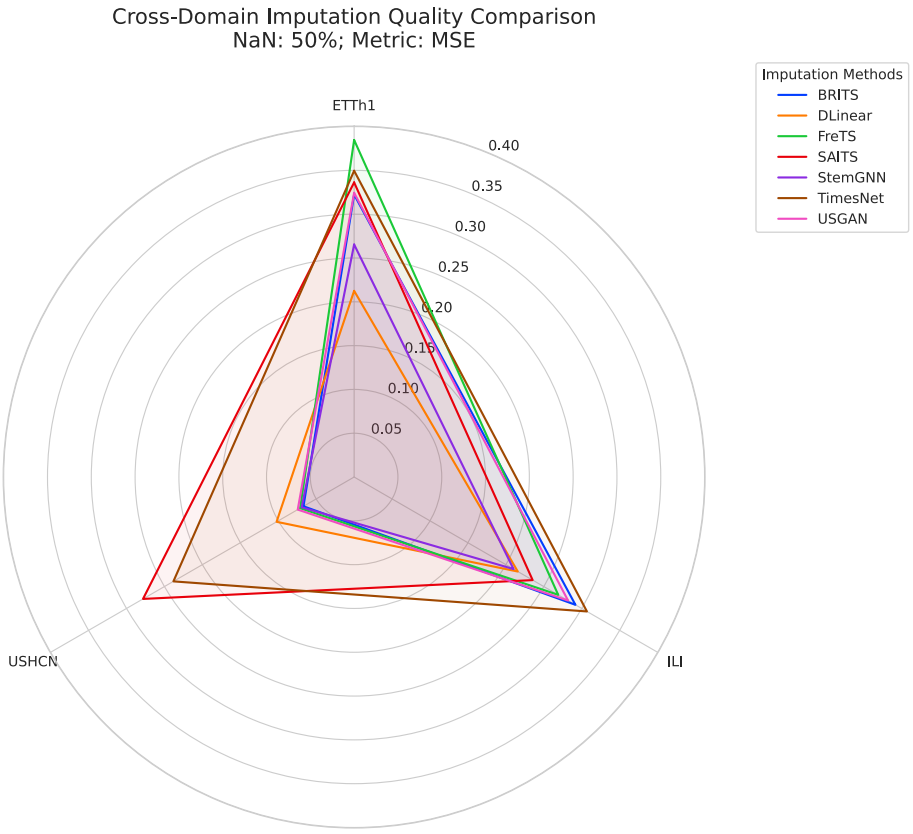


(a)

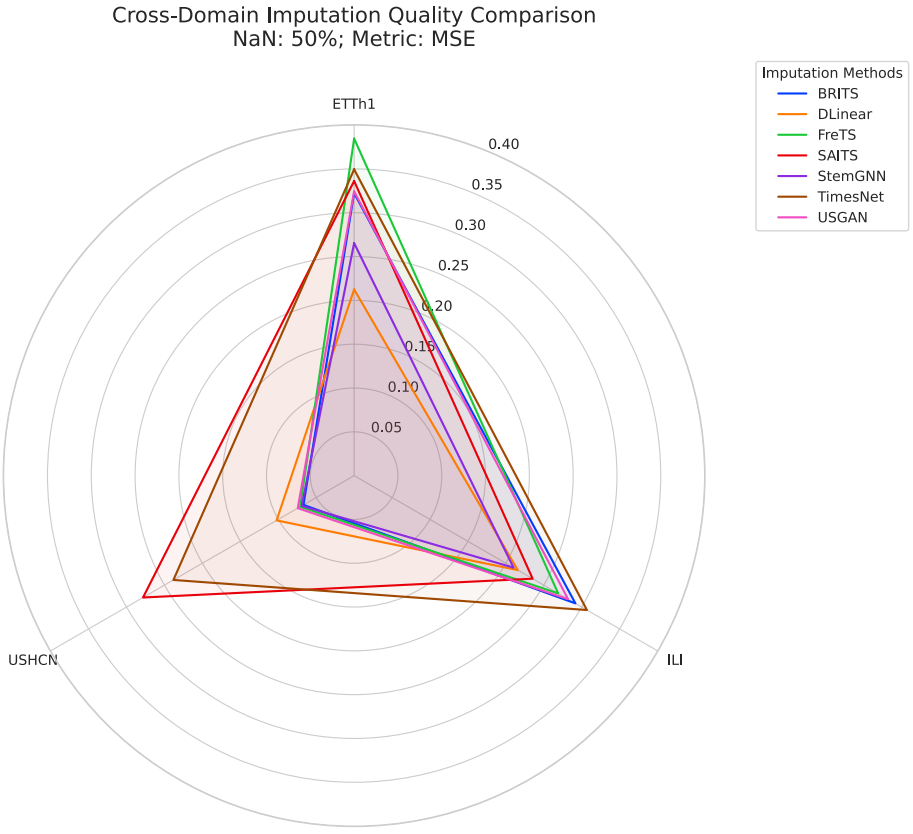


(b)

Figure A.103: Radar plot comparing imputation performance on all datasets **except** ExchangeRate across complex imputation methods with 20% missing data.

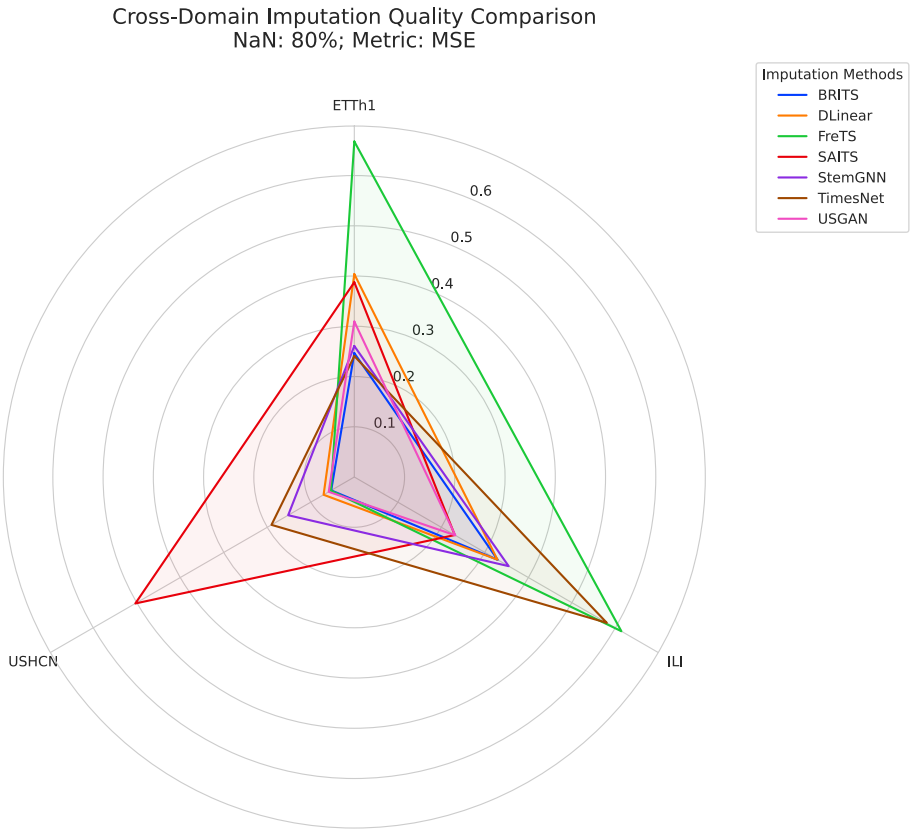


(a)

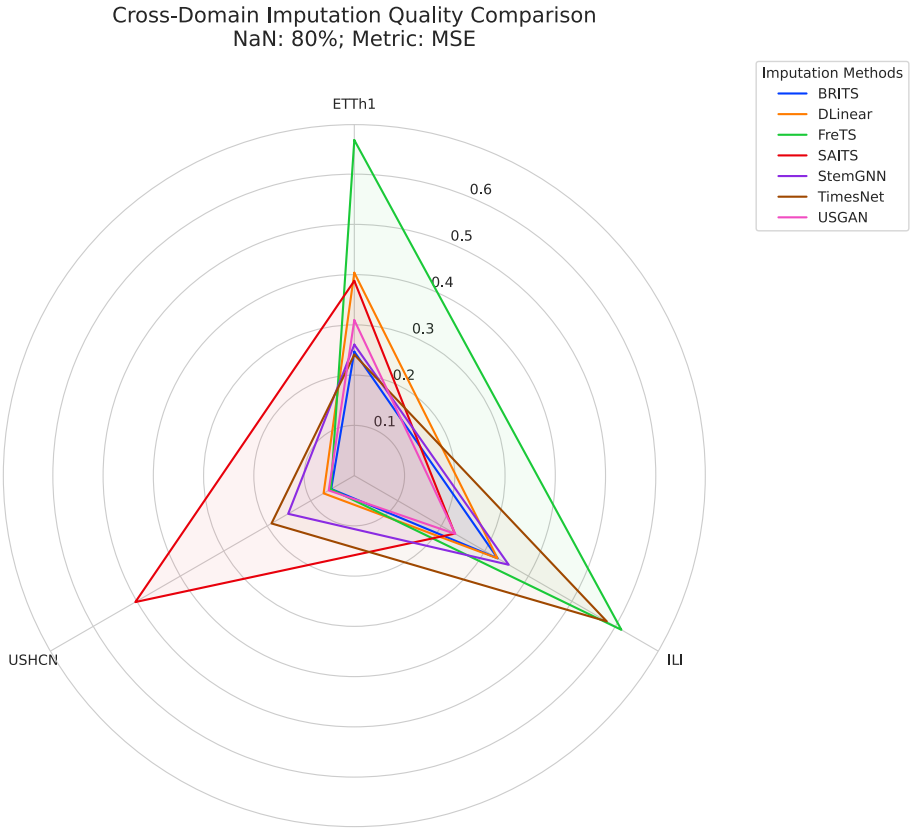


(b)

Figure A.104: Radar plot comparing imputation performance on all datasets **except** ExchangeRate across complex imputation methods with 50% missing data.



(a)

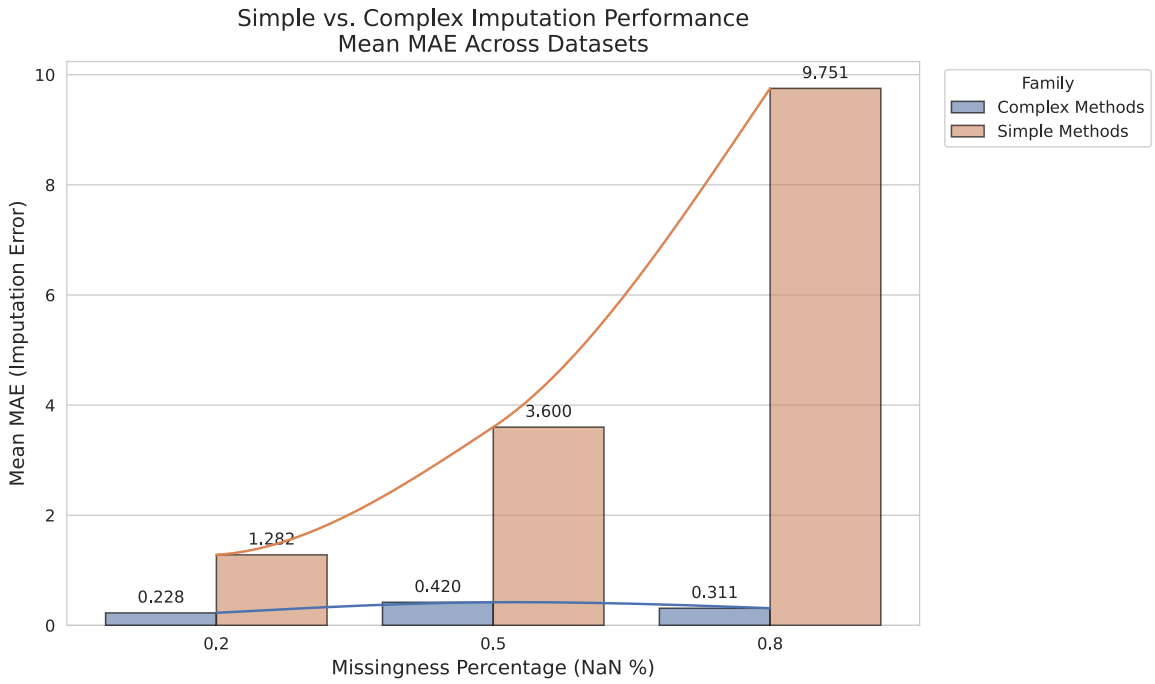


(b)

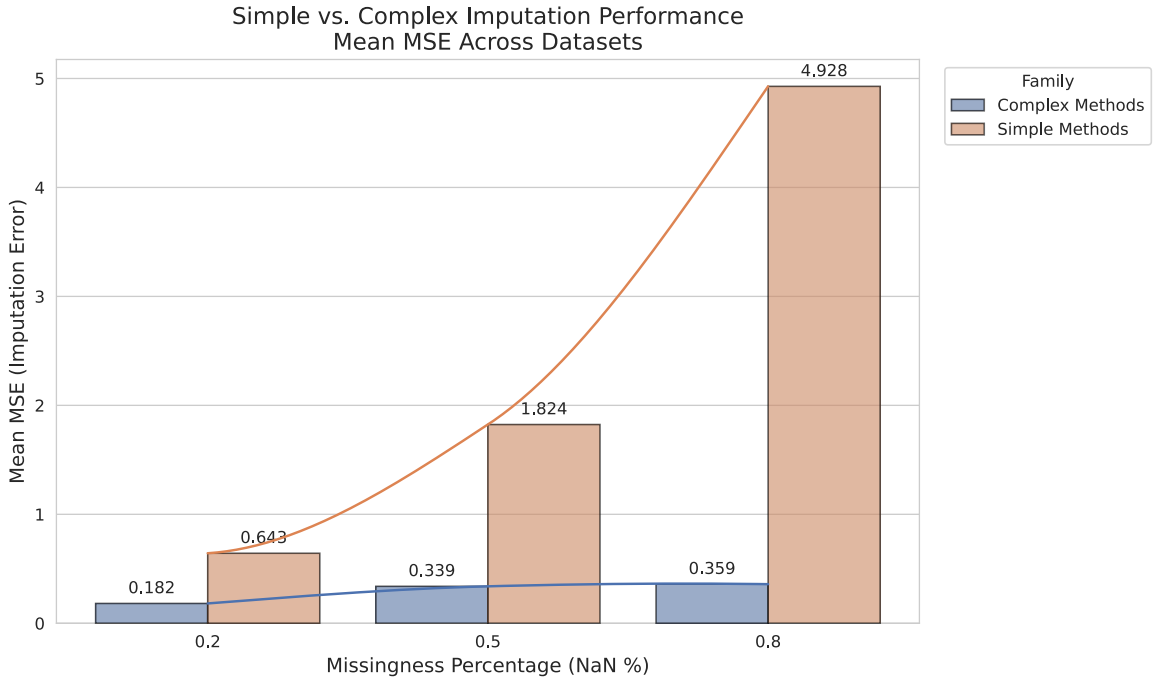
Figure A.105: Radar plot comparing imputation performance on all datasets **except** ExchangeRate across complex imputation methods with 80% missing data.

A.4.4 Stochastic Robustness Barplots

Bar charts show the mean performance across five randomness seeds ($0 - 4$), with a smoothed line over each bar to highlight the trend of architectural sensitivity to specific missingness patterns between the Simple and Complex imputation method families.

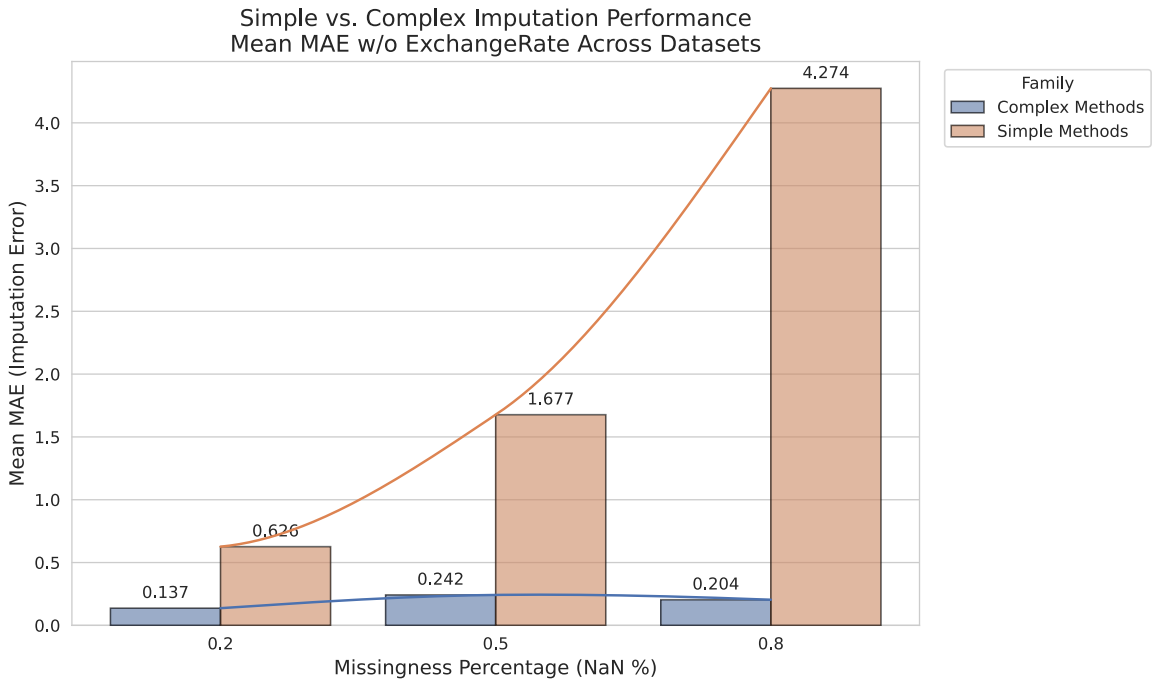


(a)

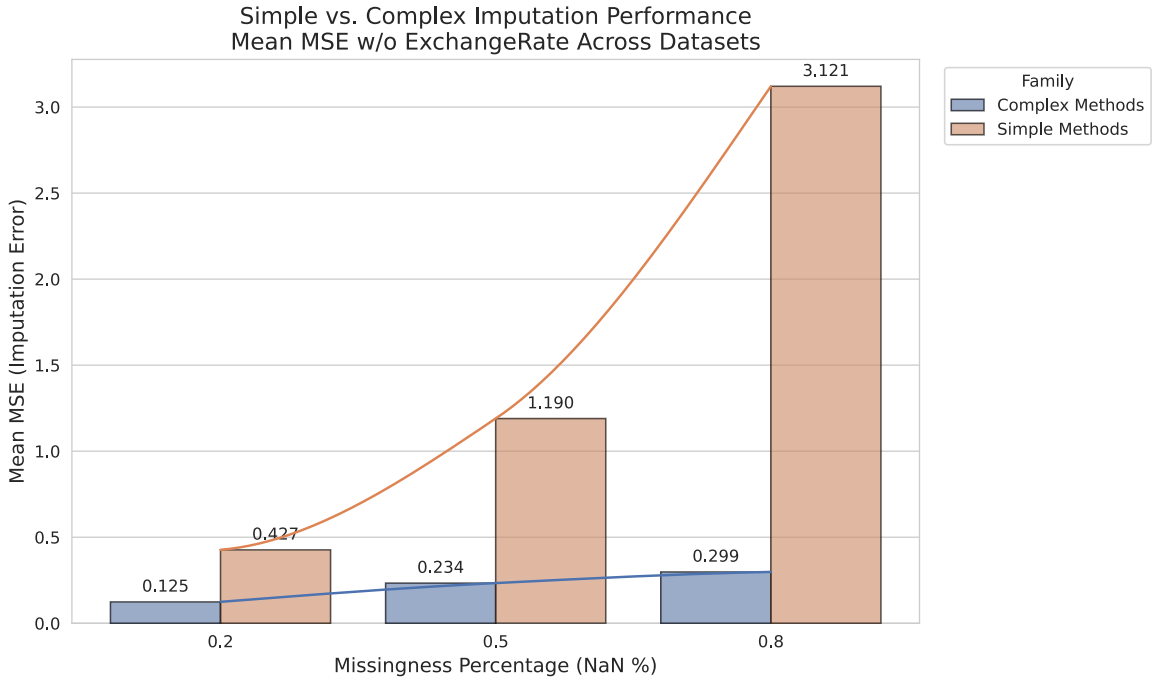


(b)

Figure A.106: Barplot comparing the imputation performance of Simple vs Complex imputation methods over all datasets.



(a)

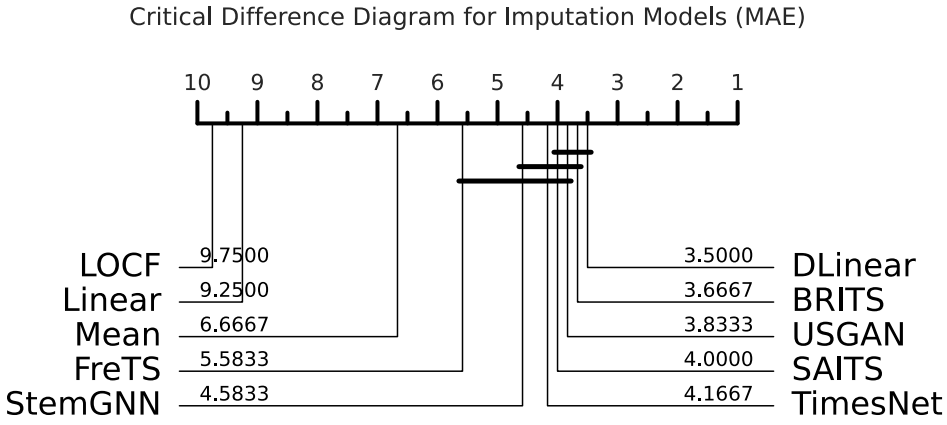


(b)

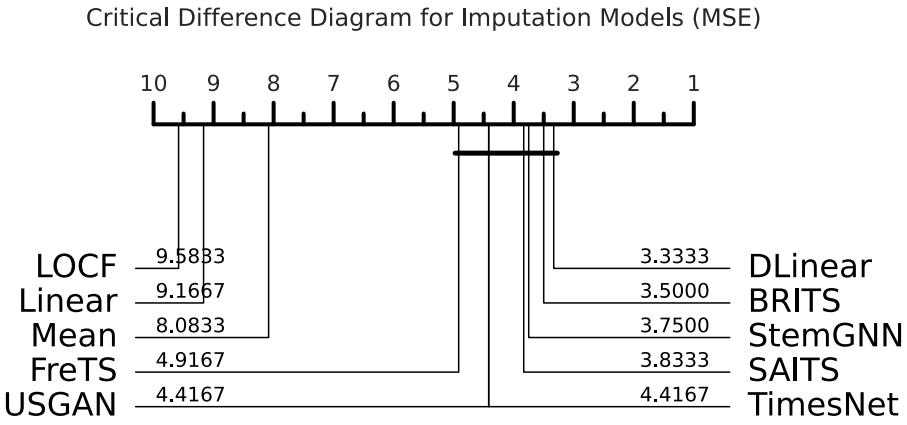
Figure A.107: Barplot comparing the imputation performance of Simple vs Complex imputation methods over all datasets **except** ExchangeRate.

A.4.5 Critical Diagrams

This section reports all Critical Difference diagrams plotted for RQ2.1 on MAE and MSE across ETTh1, ExchangeRate, USHCN and ILI datasets. CD Diagrams visualize the results of the Nemenyi post-hoc test. Models connected by a horizontal bar are not statistically different at $\alpha = 0.05$.

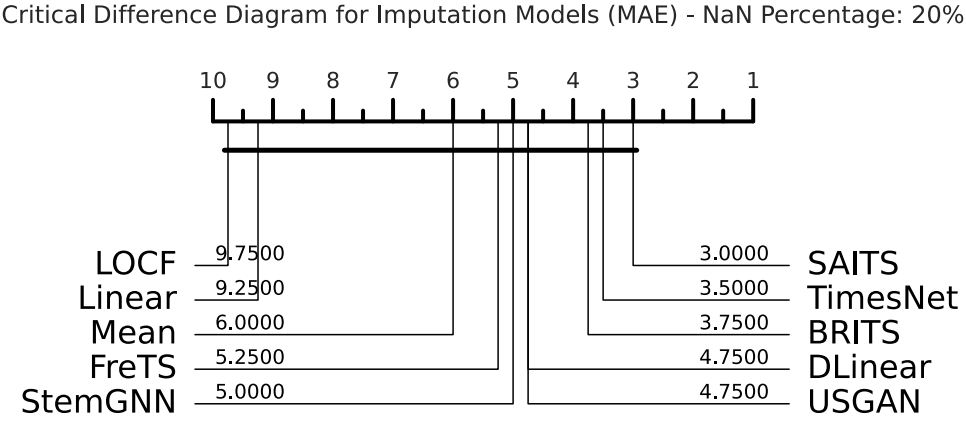


(a)

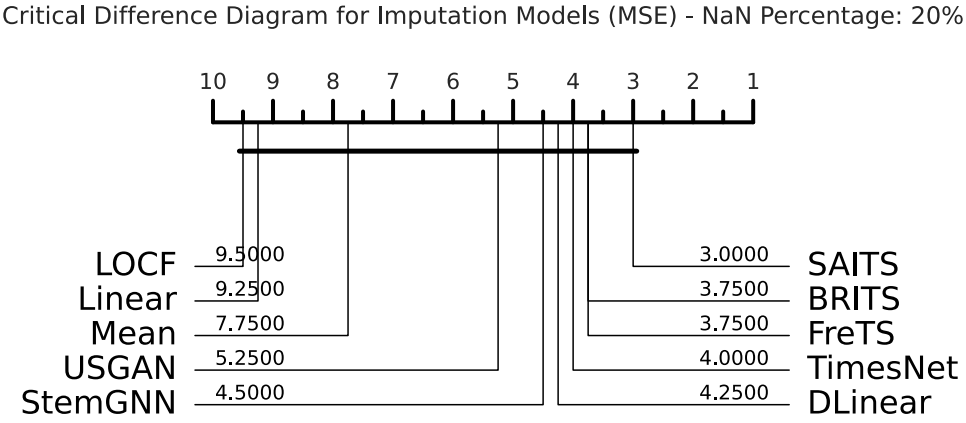


(b)

Figure A.108: Critical Difference diagrams on MAE and MSE across all NaN percentage scenarios.

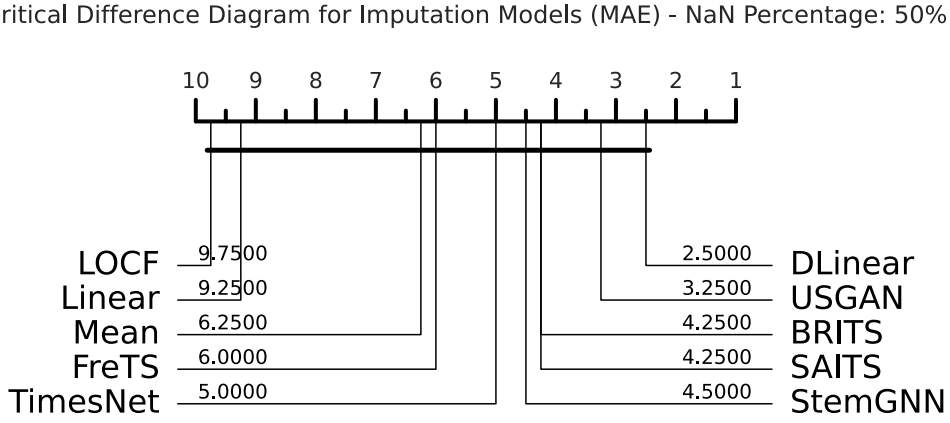


(a)

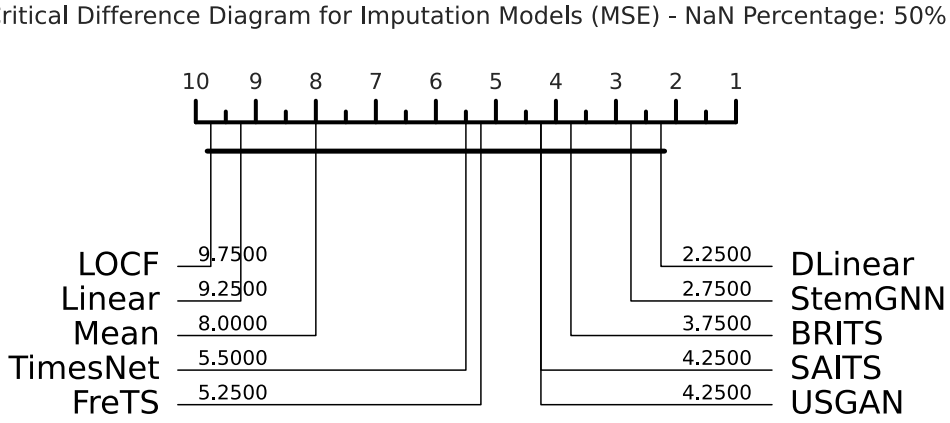


(b)

Figure A.109: Critical Difference diagrams on MAE and MSE on 20% missing data scenarios.

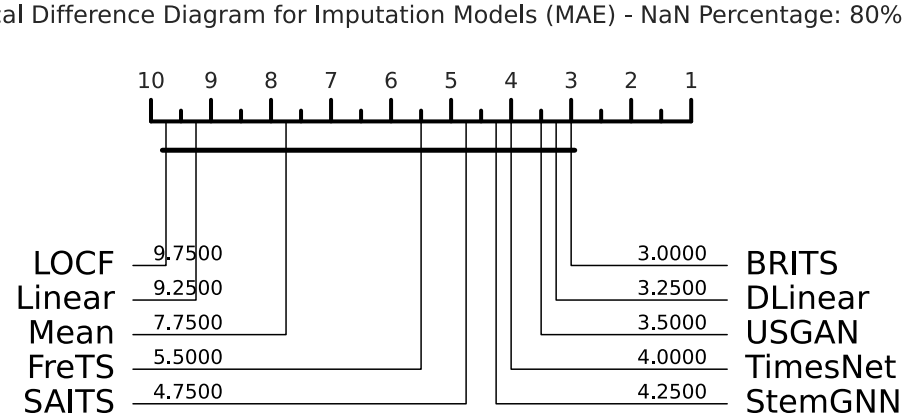


(a)

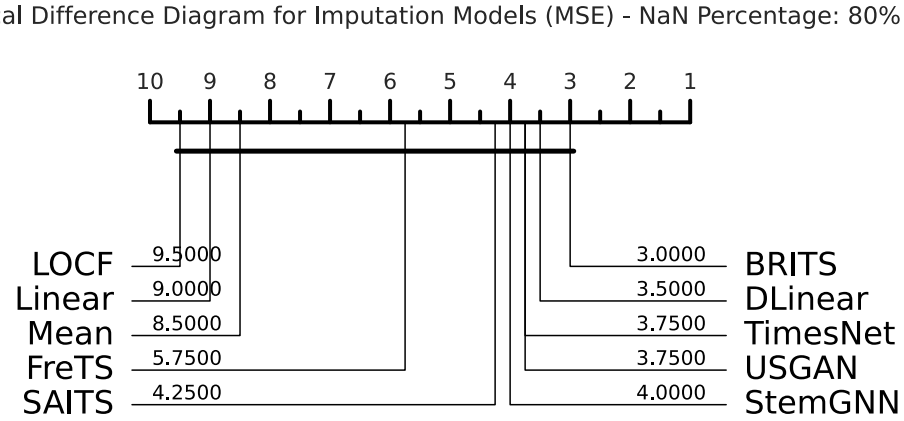


(b)

Figure A.110: Critical Difference diagrams on MAE and MSE on 50% missing data scenarios.

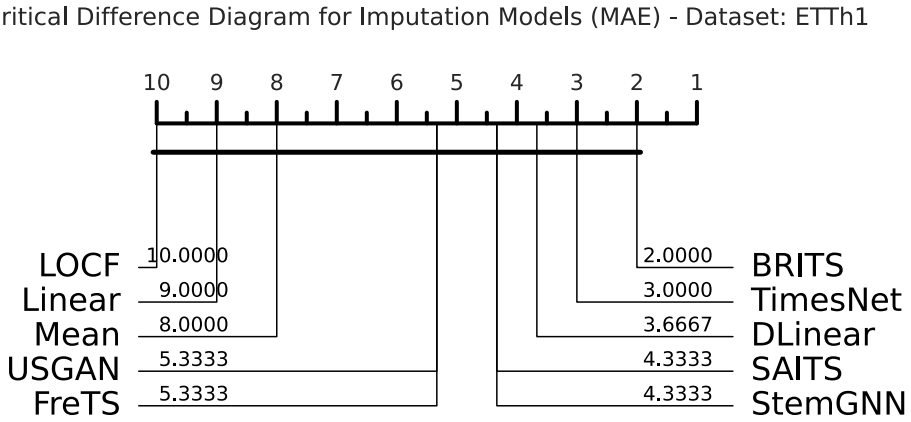


(a)

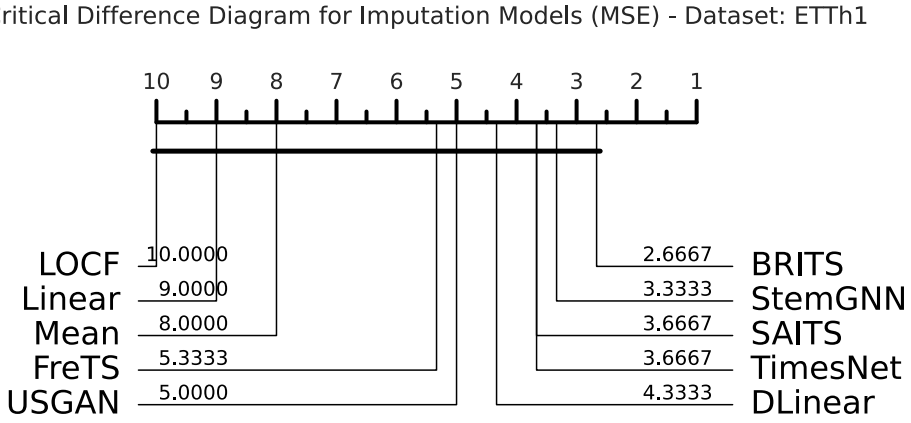


(b)

Figure A.111: Critical Difference diagrams on MAE and MSE across 80% missing data scenarios.

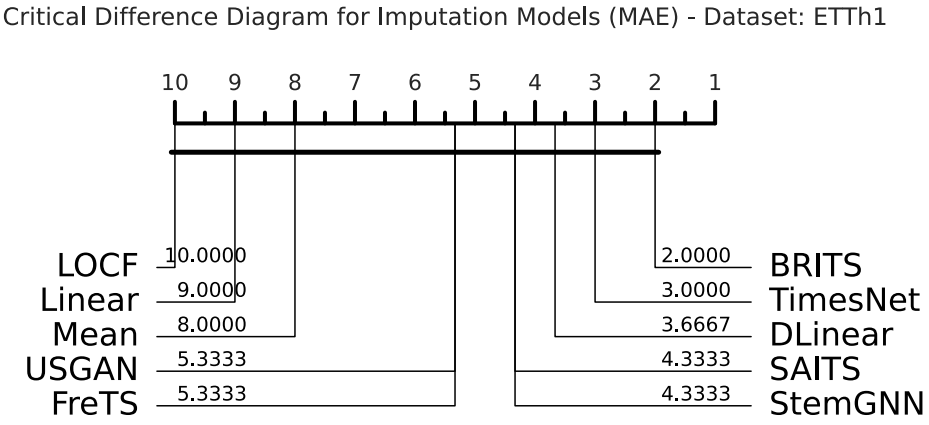


(a)

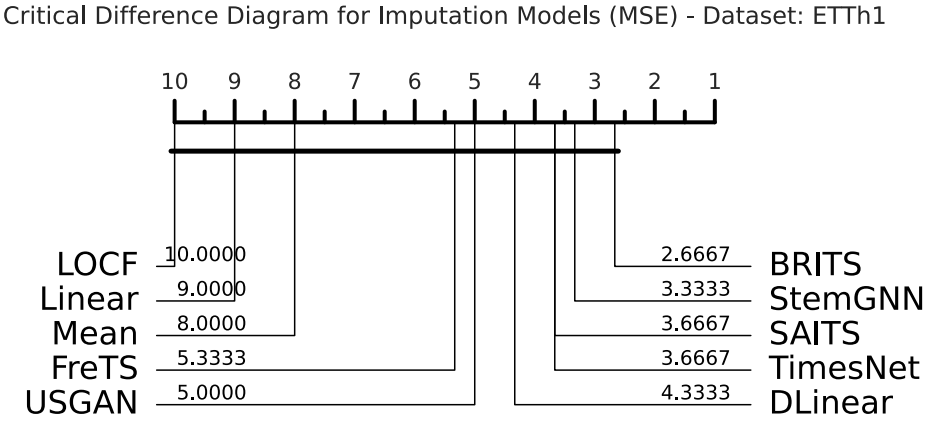


(b)

Figure A.112: Critical Difference diagrams on MAE and MSE on ETTh1 with 20% missing data.

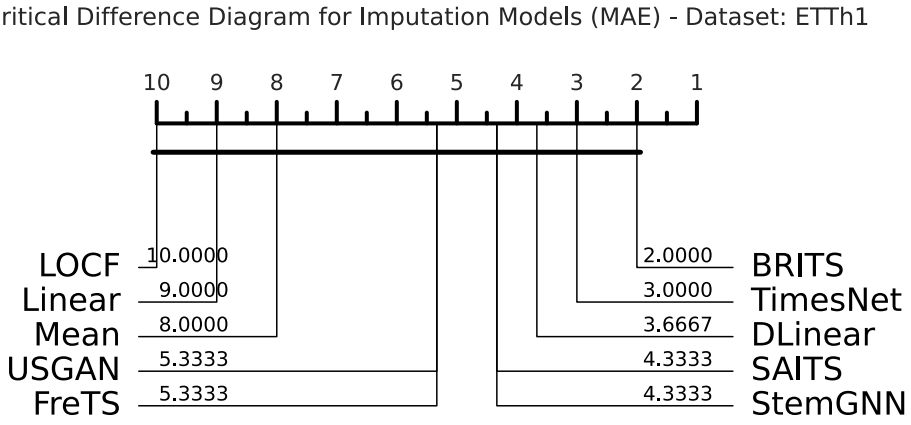


(a)

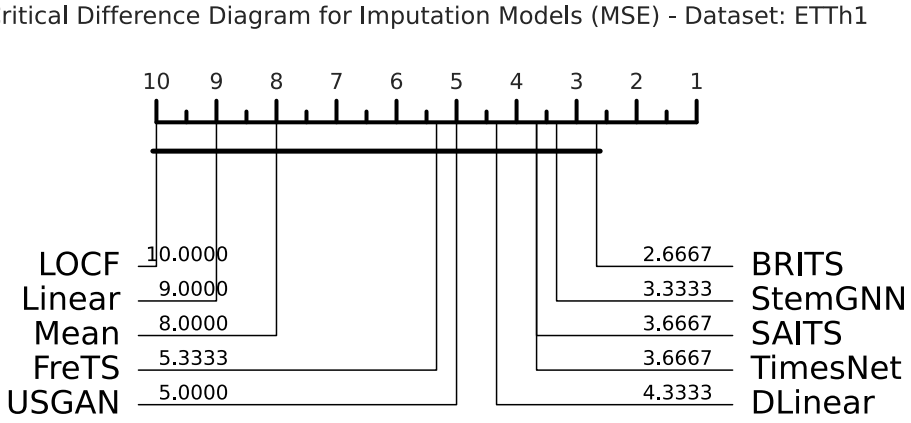


(b)

Figure A.113: Critical Difference diagrams on MAE and MSE on ETTh1 with 50% missing data.

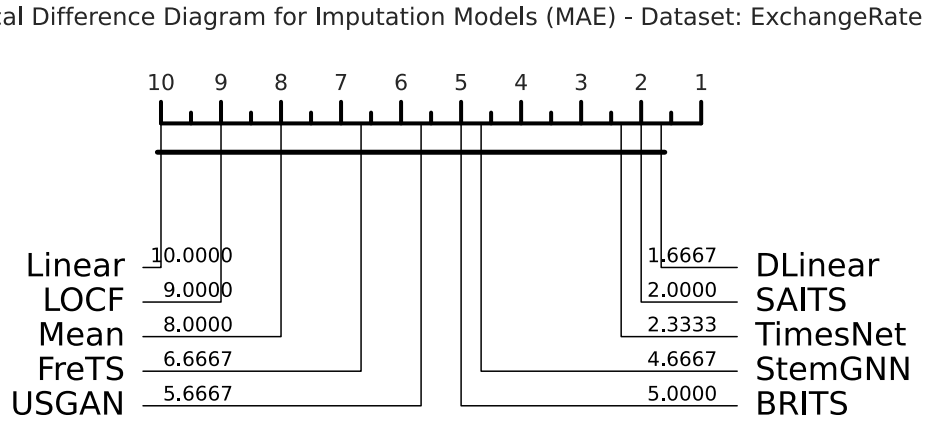


(a)

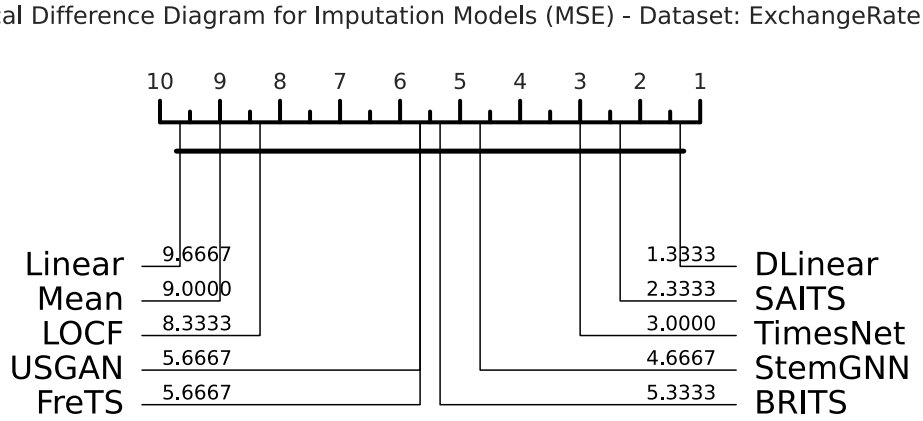


(b)

Figure A.114: Critical Difference diagrams on MAE and MSE on ETTh1 with 80% missing data.

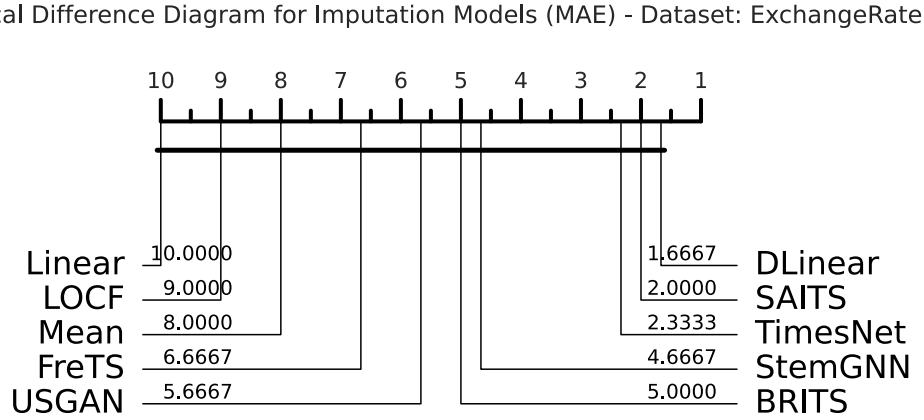


(a)

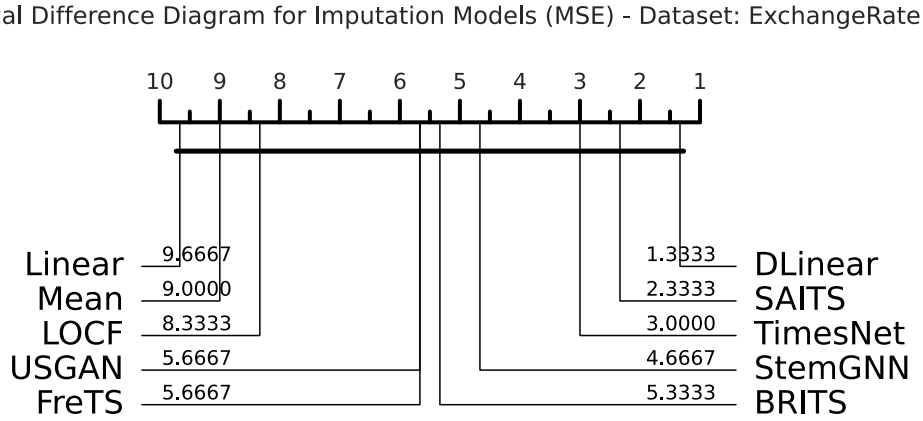


(b)

Figure A.115: Critical Difference diagrams on MAE and MSE on ExchangeRate with 20% missing data.

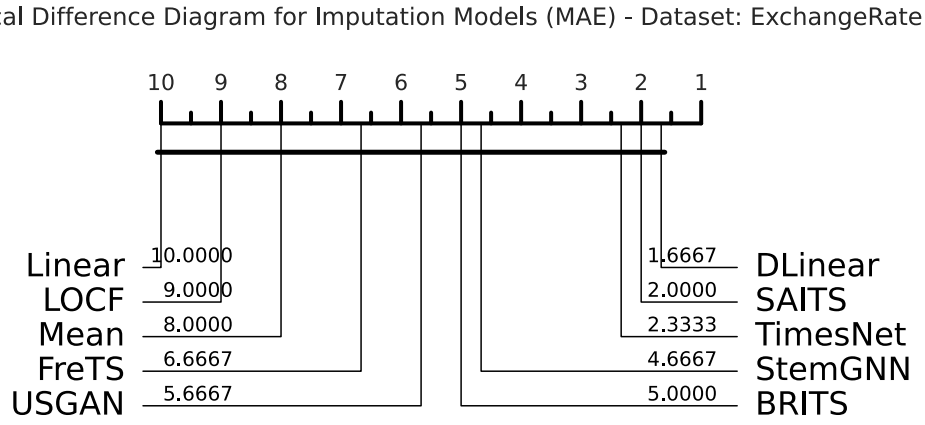


(a)

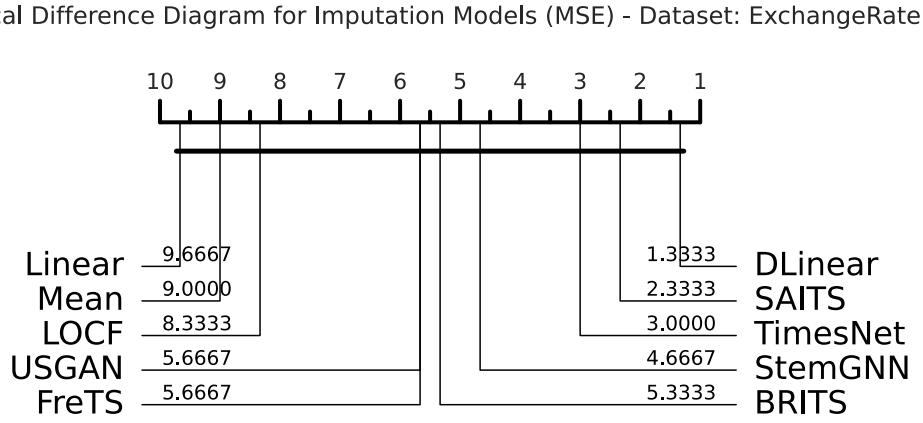


(b)

Figure A.116: Critical Difference diagrams on MAE and MSE on ExchangeRate with 50% missing data.

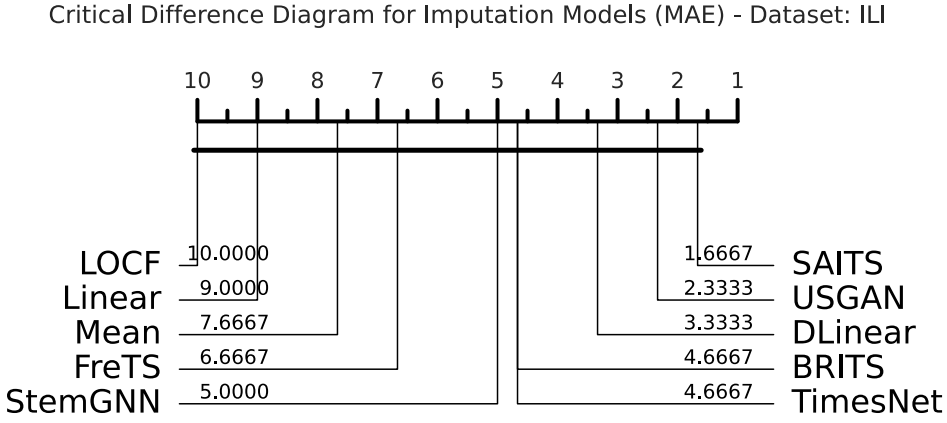


(a)

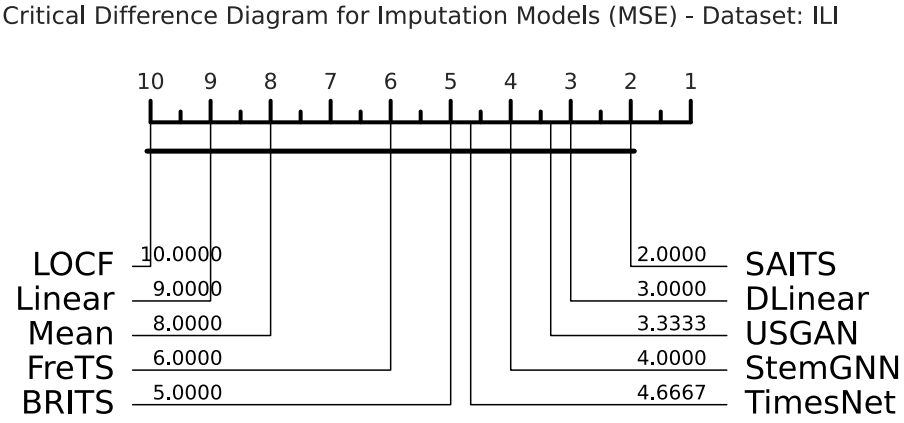


(b)

Figure A.117: Critical Difference diagrams on MAE and MSE on ExchangeRate with 80% missing data.

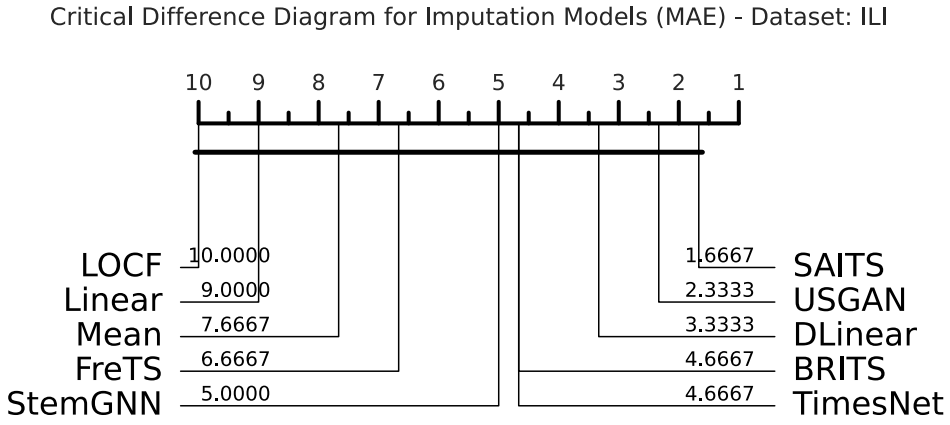


(a)

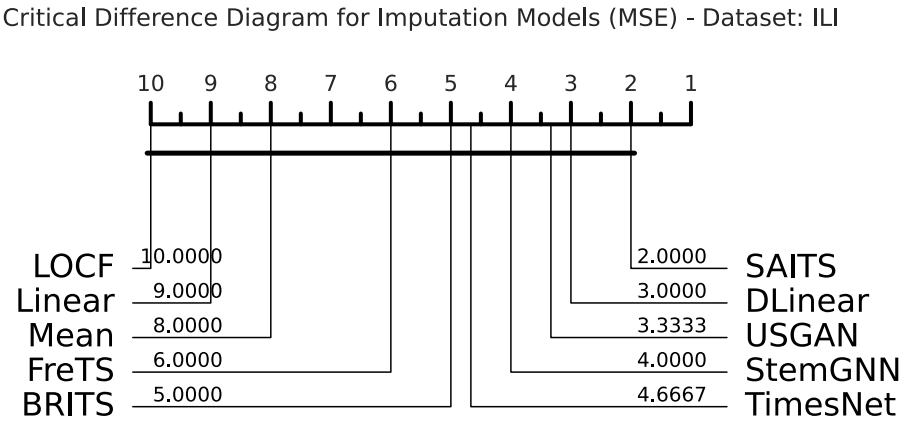


(b)

Figure A.118: Critical Difference diagrams on MAE and MSE on ILI with 20% missing data.

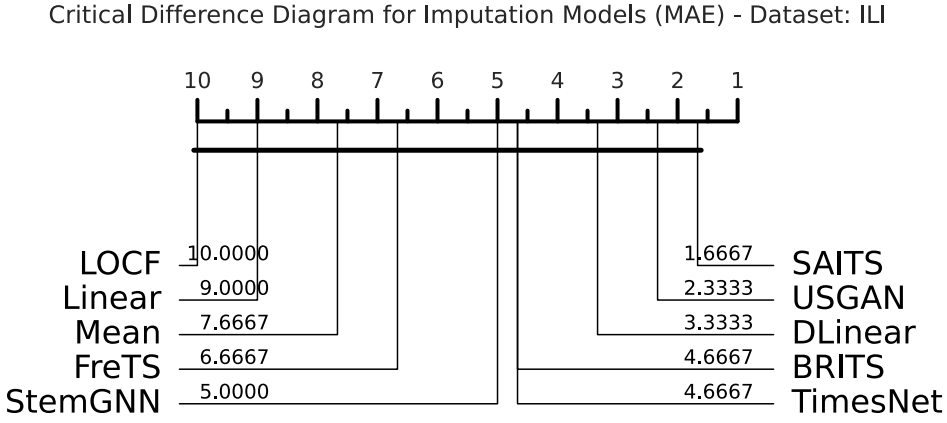


(a)

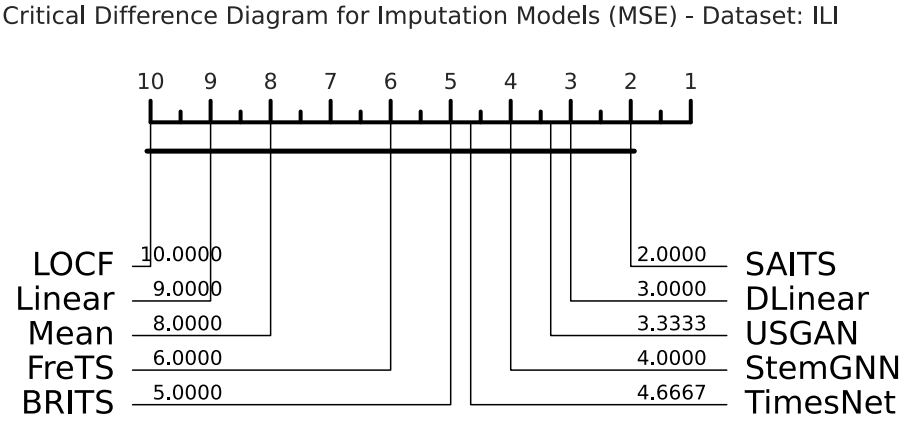


(b)

Figure A.119: Critical Difference diagrams on MAE and MSE on ILI with 50% missing data.

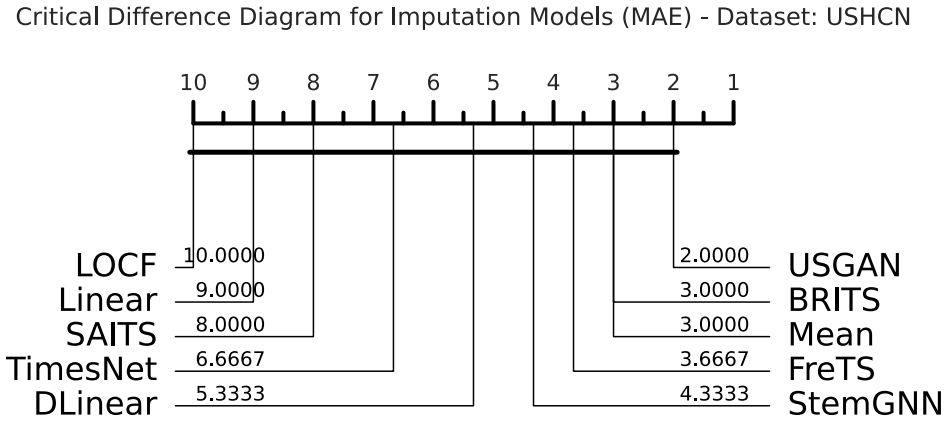


(a)

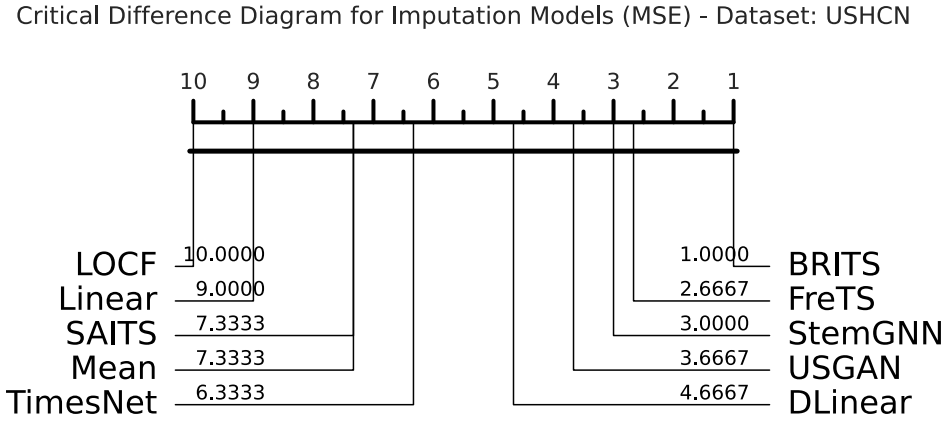


(b)

Figure A.120: Critical Difference diagrams on MAE and MSE on ILI with 80% missing data.

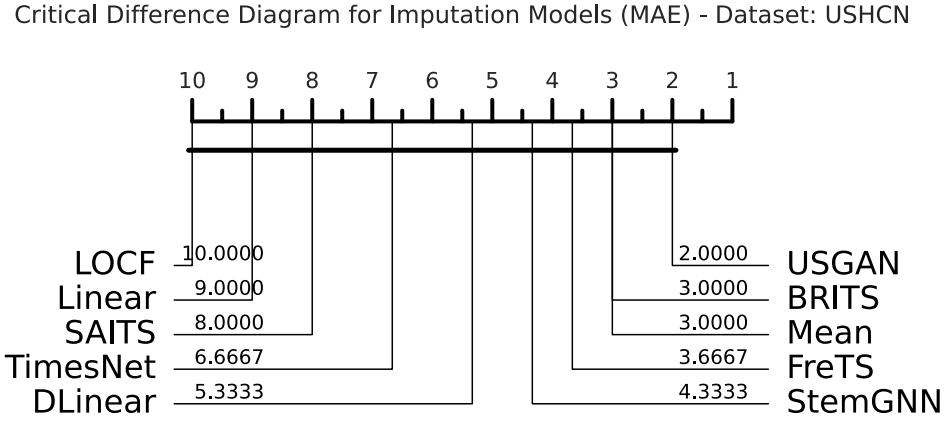


(a)

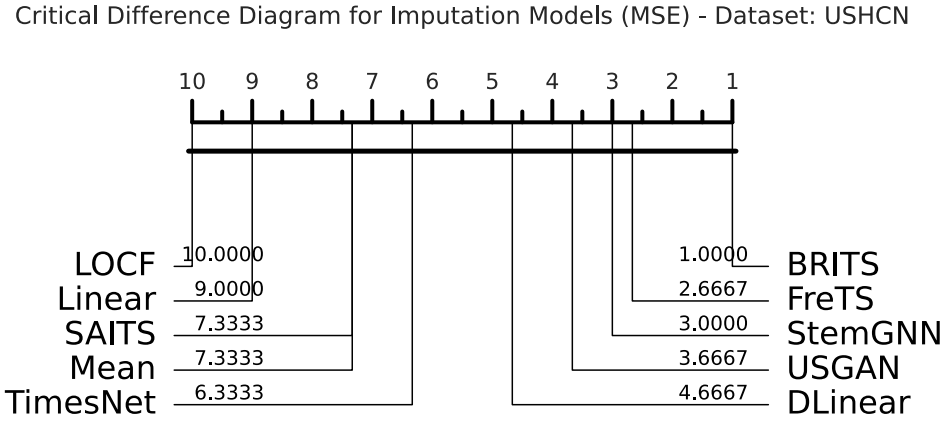


(b)

Figure A.121: Critical Difference diagrams on MAE and MSE on USHCN with 20% missing data.

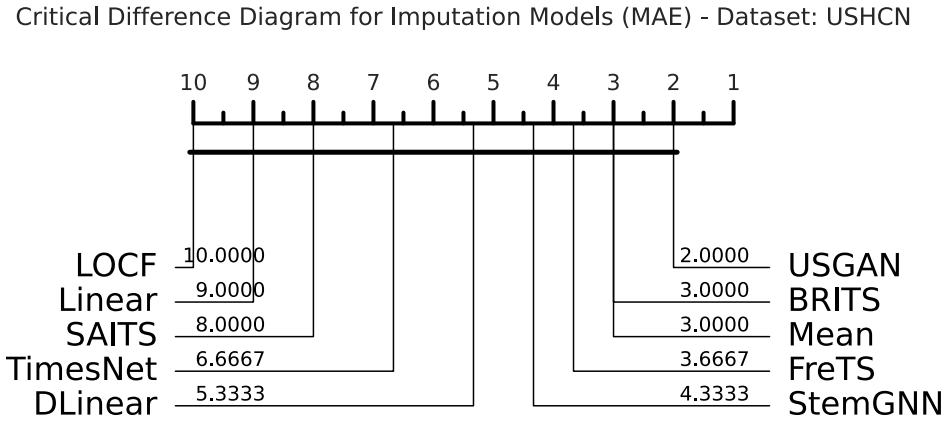


(a)

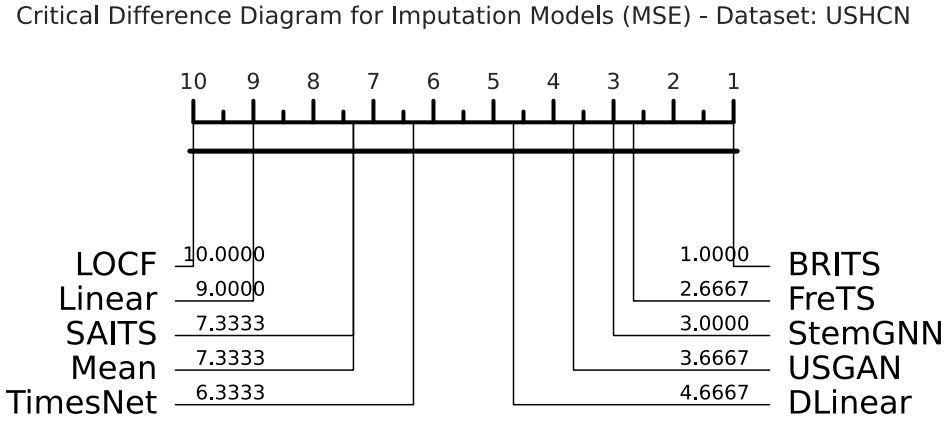


(b)

Figure A.122: Critical Difference diagrams on MAE and MSE on USHCN with 50% missing data.



(a)



(b)

Figure A.123: Critical Difference diagrams on MAE and MSE on USHCN with 80% missing data.

Bibliography

[SKG17] A. Siddiqa, A. Karim, and A. Gani, "Big Data Storage Technologies: A Survey," *Frontiers of Information Technology & Electronic Engineering*, vol. 18, no. 8, pp. 1040–1070, Aug. 2017, doi: 10.1631/FITEE.1500441.

[Yu21] S. Yu, " Data Processing and Development of Big Data System: A Survey ," in *Advances in Artificial Intelligence and Security*, X. Sun, X. Zhang, Z. Xia, and E. Bertino, Eds., Cham: Springer International Publishing, 2021, pp. 420–431.

[Bri15] D. R. Brillinger, "Time Series: General," *International Encyclopedia of the Social & Behavioral Sciences (Second Edition)*. Elsevier, Oxford, pp. 341–346, 2015. doi: https://doi.org/10.1016/B978-0-08-097086-8.42084-2.

[NIS03] NIST, *NIST/SEMATECH e-Handbook of Statistical Methods*,. NIST, 2003, p. . doi: https://doi.org/10.18434/M32189.

[Wil16] G. T. Wilson, " Time Series Analysis: Forecasting and Control, 5th Edition, by George E. P. Box, Gwilym M. Jenkins, Gregory C. Reinsel and Greta M. Ljung, 2015. Published by John Wiley and Sons Inc., Hoboken, New Jersey, Pp. 712. ISBN: 978-1-118-67502-1 ," *Journal of Time Series Analysis*, vol. 37, no. 5, pp. 709–711, Sept. 2016, doi: 10.1111/jtsa.12194.

[WYY15] L. Wu, L. Yuan, and J. You, " Survey of Large-Scale Data Management Systems for Big Data Applications ," *Journal of Computer Science and Technology*, vol. 30, no. 1, pp. 163–183, Jan. 2015, doi: 10.1007/s11390-015-1511-8.

[KBY17] A. Kumar, M. Boehm, and J. Yang, " Data Management in Machine Learning: Challenges, Techniques, and Systems ," in *Proceedings of the 2017 ACM International Conference on Management of Data*, in SIGMOD '17. New York, NY, USA: Association for Computing Machinery, 2017, pp. 1717–1722. doi: 10.1145/3035918.3054775.

[TSM18] Q.-C. To, J. Soto, and V. Markl, "A Survey of State Management in Big Data Processing Systems," *The VLDB Journal*, vol. 27, no. 6, pp. 847–872, Dec. 2018, doi: 10.1007/s00778-018-0514-9.

[Ben+25] R. Benassi, F. Del Buono, G. Guiduzzi, and F. Guerra, " Forecasting Irregularly Sampled Time Series with Transformer Encoders ," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, Springer, 2025, pp. 201–217.

[Mas+25] C. Masciulli *et al.*, " Interpretable Clustering of PS-InSAR Time Series for Ground Deformation Detection ," *Computers & Geosciences*, p. 105959, 2025.

[ANA22] O. A. Alcántara Francia, M. Nunez-del-Prado, and H. Alatrista-Salas, " Survey of Text Mining Techniques Applied to Judicial Decisions Prediction ," *Applied Sciences*, vol. 12, no. 20, p. 10200, Jan. 2022, doi: 10.3390/app122010200.

[Bar+23] A. Baraldi *et al.*, "Interpretable Entity Matching with WYM," in *CEUR WORKSHOP PROCEEDINGS*, CEUR-WS, 2023, pp. 101–109.

[Pag+24] M. Paganelli *et al.*, "How Transformers Are Revolutionizing Entity Matching," in *CEUR WORKSHOP PROCEEDINGS*, CEUR-WS, 2024, pp. 662–670.

[CS23] G. Chiarot and C. Silvestri, "Time Series Compression Survey," *ACM Comput. Surv.*, vol. 55, no. 10, pp. 198:1–198:32, Feb. 2023, doi: 10.1145/3560814.

[HAB22] H. Hewamalage, K. Ackermann, and C. Bergmeir, " Forecast Evaluation for Data Scientists: Common Pitfalls and Best Practices ," no. arXiv:2203.10716. arXiv, Apr. 2022. doi: 10.48550/arXiv.2203.10716.

[CTM20] V. Cerqueira, L. Torgo, and I. Mozetic, " Evaluating Time Series Forecasting Models: An Empirical Study on Performance Estimation Methods ," *Machine Learning*, vol. 109, no. 11, pp. 1997–2028, Nov. 2020, doi: 10.1007/s10994-020-05910-7.

[Qiu+24] X. Qiu *et al.*, " TFB: Towards Comprehensive and Fair Benchmarking of Time Series Forecasting Methods ," *Proceedings of the VLDB Endowment*, vol. 17, no. 9, pp. 2363–2377, May 2024, doi: 10.14778/3665844.3665863.

[Shc+25] O. Shchur *et al.*, "Fev-Bench: A Realistic Benchmark for Time Series Forecasting," no. arXiv:2509.26468. arXiv, Sept. 2025. doi: 10.48550/arXiv.2509.26468.

[Mey+25] M. Meyer, S. Kaltenpoth, K. Zalipski, and O. Müller, "Time Series Foundation Models: Benchmarking Challenges and Requirements," no. arXiv:2510.13654. arXiv, Oct. 2025. doi: 10.48550/arXiv.2510.13654.

[Li+23] Z. Li, S. Qi, Y. Li, and Z. Xu, " Revisiting Long-term Time Series Forecasting: An Investigation on Linear Mapping ," no. arXiv:2305.10721. arXiv, May 2023a. doi: 10.48550/arXiv.2305.10721.

[Zho+25] Y. Zhou, Y. Wang, S. Goel, and A. R. Zhang, "Why Do Transformers Fail to Forecast Time Series In-Context?," no. arXiv:2510.09776. arXiv, Oct. 2025a. doi: 10.48550/arXiv.2510.09776.

[Gol+24] M. Goldblum, M. Finzi, K. Rowan, and A. G. Wilson, " The No Free Lunch Theorem, Kolmogorov Complexity, and the Role of Inductive Biases in Machine Learning ," no. arXiv:2304.05366. arXiv, June 2024. doi: 10.48550/arXiv.2304.05366.

[LPK22] P. Liakos, K. Papakonstantinopoulou, and Y. Kotidis, " Chimp: Efficient Lossless Floating Point Compression for Time Series Databases ," *Proc. VLDB Endow.*, vol. 15, no. 11, pp. 3058–3070, 2022, doi: 10.14778/3551793.3551852.

[Gue+25] A. Guerra, G. Vinciguerra, A. Boffa, and P. Ferragina, "Learned Compression of Nonlinear Time Series with Random Access," in *2025 IEEE 41st International Conference on Data Engineering (ICDE)*, May 2025, pp. 1579–1592. doi: 10.1109/ICDE65448.2025.00122.

[CSW23] J. Cui, X. Shen, and S. Wen, " A Survey on Legal Judgment Prediction: Datasets, Metrics, Models and Challenges ," *IEEE Access*, vol. 11, pp. 102050–102071, 2023, doi: 10.1109/ACCESS.2023.3317083.

[Cha+25] C. Chang *et al.*, " Time-IMM: A Dataset and Benchmark for Irregular Multimodal Multivariate Time Series ," no. arXiv:2506.10412. arXiv, June 2025. doi: 10.48550/arXiv.2506.10412.

[Gha+24] E. Ghaderpour, P. Mazzanti, F. Bozzano, and G. S. Mugnozza, " Ground Deformation Monitoring via PS-InSAR Time Series: An Industrial Zone in Sacco River

Valley, Central Italy ," *Remote Sensing Applications: Society and Environment*, vol. 34, p. 101191, 2024a, doi: 10.1016/j.rsase.2024.101191.

[WM97] D. Wolpert and W. Macready, "No Free Lunch Theorems for Optimization," *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 67–82, Apr. 1997, doi: 10.1109/4235.585893.

[No 12] "No Free Lunch Theorem: A Review." Aug. 2012.

[RF23] M. Rashki and M. G. R. Faes, "No-Free-Lunch Theorems for Reliability Analysis," *ASCE-ASME Journal of Risk and Uncertainty in Engineering Systems, Part A: Civil Engineering*, vol. 9, no. 3, p. 4023019, Sept. 2023, doi: 10.1061/AJRUA6.RU-ENG-1015.

[Bro21] J. Brownlee, "No Free Lunch Theorem for Machine Learning." Feb. 2021.

[Zen+22] A. Zeng, M. Chen, L. Zhang, and Q. Xu, "Are Transformers Effective for Time Series Forecasting?," no. arXiv:2205.13504. arXiv, Aug. 2022. doi: 10.48550/arXiv.2205.13504.

[VW24] J. de Vilmarest and N. Werge, " An Adaptive Volatility Method for Probabilistic Forecasting and Its Application to the M6 Financial Forecasting Competition ," no. arXiv:2303.01855. arXiv, June 2024. doi: 10.48550/arXiv.2303.01855.

[SRP24] L. Schmid, M. Roidl, and M. Pauly, " Comparing Statistical and Machine Learning Methods for Time Series Forecasting in Data-Driven Logistics – a Simulation Study ," no. arXiv:2303.07139. arXiv, June 2024. doi: 10.48550/arXiv.2303.07139.

[Kon+25] X. Kong *et al.*, "Deep Learning for Time Series Forecasting: A Survey," *International Journal of Machine Learning and Cybernetics*, vol. 16, no. 7–8, pp. 5079–5112, Aug. 2025, doi: 10.1007/s13042-025-02560-w.

[SFG19] D. Salinas, V. Flunkert, and J. Gasthaus, " DeepAR: Probabilistic Forecasting with Autoregressive Recurrent Networks ," no. arXiv:1704.04110. arXiv, Feb. 2019. doi: 10.48550/arXiv.1704.04110.

[Ore+20] B. N. Oreshkin, D. Carpov, N. Chapados, and Y. Bengio, " N-BEATS: Neural Basis Expansion Analysis for Interpretable Time Series Forecasting ," no. arXiv:1905.10437. arXiv, Feb. 2020. doi: 10.48550/arXiv.1905.10437.

[LZ21] B. Lim and S. Zohren, "Time-Series Forecasting with Deep Learning: A Survey," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 379, no. 2194, p. 20200209, Feb. 2021, doi: 10.1098/rsta.2020.0209.

[FB20] C. Fry and M. Brundage, "The M4 Forecasting Competition – a Practitioner's View," *International Journal of Forecasting*, vol. 36, no. 1, pp. 156–160, Jan. 2020, doi: 10.1016/j.ijforecast.2019.02.013.

[EY25] T. Ekiz Yılmaz and G. Yapar, " A Robust Hybrid Forecasting Framework for the M3 and M4 Competitions: Combining ARIMA and Ata Models with Performance-Based Model Selection ," *Applied Sciences*, vol. 15, no. 9552, 2025, doi: 10.3390/app15179552.

[Zho+21] H. Zhou *et al.*, " Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting ," no. arXiv:2012.07436. arXiv, Mar. 2021. doi: 10.48550/arXiv.2012.07436.

[Wu+22] H. Wu, J. Xu, J. Wang, and M. Long, " Autoformer: Decomposition Transformers with Auto-Correlation for Long-Term Series Forecasting ," no. arXiv:2106.13008. arXiv, Jan. 2022. doi: 10.48550/arXiv.2106.13008.

[Zho+22] T. Zhou, Z. Ma, Q. Wen, X. Wang, L. Sun, and R. Jin, " FEDformer: Frequency Enhanced Decomposed Transformer for Long-Term Series Forecasting ," no. arXiv:2201.12740. arXiv, June 2022. doi: 10.48550/arXiv.2201.12740.

[RKN25] S. T. H. Rizvi, N. Kanwal, and M. Naeem, " Bridging Simplicity and Sophistication Using GLinear: A Novel Architecture for Enhanced Time Series Prediction ," no. arXiv:2501.01087. arXiv, Sept. 2025. doi: 10.48550/arXiv.2501.01087.

[Hua+23] Y. Huang, Z. Zhou, Z. Wang, X. Zhi, and X. Liu, " TimesNet-PM2.5: Interpretable TimesNet for Disentangling Intraperiod and Interperiod Variations in PM2.5 Prediction ," *Atmosphere*, vol. 14, no. 11, p. 1604, Nov. 2023, doi: 10.3390/atmos14111604.

[Wu+23] H. Wu, T. Hu, Y. Liu, H. Zhou, J. Wang, and M. Long, " TimesNet: Temporal 2D-variation Modeling for General Time Series Analysis ," no. arXiv:2210.02186. arXiv, Apr. 2023. doi: 10.48550/arXiv.2210.02186.

[Sze+14] C. Szegedy *et al.*, "Going Deeper with Convolutions," no. arXiv:1409.4842. arXiv, Sept. 2014. doi: 10.48550/arXiv.1409.4842.

[Ye+25] J. Ye, Y. Yu, W. Zhang, L. Wang, J. Li, and F. Tsung, " Empowering Time Series Analysis with Foundation Models: A Comprehensive Survey ," no. arXiv:2405.02358. arXiv, Sept. 2025. doi: 10.48550/arXiv.2405.02358.

[Mil+24] J. A. Miller *et al.*, " A Survey of Deep Learning and Foundation Models for Time Series Forecasting ," no. arXiv:2401.13912. arXiv, Jan. 2024. doi: 10.48550/arXiv.2401.13912.

[Gru+24] N. Gruver, M. Finzi, S. Qiu, and A. G. Wilson, "Large Language Models Are Zero-Shot Time Series Forecasters," no. arXiv:2310.07820. arXiv, Aug. 2024. doi: 10.48550/arXiv.2310.07820.

[Kot+25] S. R. K. Kottapalli *et al.*, "Foundation Models for Time Series: A Survey," no. arXiv:2504.04011. arXiv, Apr. 2025. doi: 10.48550/arXiv.2504.04011.

[Das+24] A. Das, W. Kong, R. Sen, and Y. Zhou, "A Decoder-Only Foundation Model for Time-Series Forecasting," in *Forty-First International Conference on Machine Learning*, June 2024.

[Liu+25] C. Liu *et al.*, "Moirai 2.0: When Less Is More for Time Series Forecasting," no. arXiv:2511.11698. arXiv, Nov. 2025a. doi: 10.48550/arXiv.2511.11698.

[MSA20] S. Makridakis, E. Spiliotis, and V. Assimakopoulos, *The M5 Accuracy Competition: Results, Findings and Conclusions*. 2020.

[SPA23] E. Spiliotis, F. Petropoulos, and V. Assimakopoulos, "On the Disagreement of Forecasting Model Selection Criteria," *Forecasting*, vol. 5, no. 2, pp. 487–498, June 2023, doi: 10.3390/forecast5020027.

[God+25] C. Goddard, L. M. Smith, V. Ngampruetikorn, and D. J. Schwab, "When can in-context learning generalize out of task distribution?," in *Forty-second International Conference on Machine Learning*, 2025. [Online]. Available: https://openreview.net/forum?id=YKyza9lrv4

[MSA22] S. Makridakis, E. Spiliotis, and V. Assimakopoulos, "M5 Accuracy Competition: Results, Findings, and Conclusions," *International Journal of Forecasting*, Jan. 2022, doi: 10.1016/j.ijforecast.2021.11.013.

[SP25] L. Steinmeister and M. Pauly, " Iterative Trace Minimization for the Reconciliation of Very Short Hierarchical Time Series ." [Online]. Available: https://arxiv.org/abs/2409.18550

[Spr+24] O. Sprangers, W. Wadman, S. Schelter, and M. de Rijke, "Hierarchical Forecasting at Scale," no. arXiv:2310.12809. arXiv, Feb. 2024. doi: 10.48550/arXiv.2310.12809.

[Mak+24] S. Makridakis, E. Spiliotis, R. Hollyman, F. Petropoulos, N. Swanson, and A. Gaba, " The M6 Forecasting Competition: Bridging the Gap between Forecasting and Investment Decisions ," *International Journal of Forecasting*, vol. 41, Nov. 2024, doi: 10.1016/j.ijforecast.2024.11.002.

[KN22] S. Kapoor and A. Narayanan, "Leakage and the Reproducibility Crisis in ML-based Science," no. arXiv:2207.07048. arXiv, July 2022. doi: 10.48550/arXiv.2207.07048.

[AA25] S. Albelali and M. Ahmed, " Hidden Leaks in Time Series Forecasting: How Data Leakage Affects LSTM Evaluation across Configurations and Validation Strategies ," no. arXiv:2512.06932. arXiv, Dec. 2025. doi: 10.48550/arXiv.2512.06932.

[Kim+25] J. Kim, H. Kim, H. Kim, D. Lee, and S. Yoon, " A Comprehensive Survey of Deep Learning for Time Series Forecasting: Architectural Diversity and Open Challenges ," no. arXiv:2411.05793. arXiv, May 2025. doi: 10.48550/arXiv.2411.05793.

[Tas00] L. Tashman, "Out-of-Sample Tests of Forecasting Accuracy: An Analysis and Review," *International Journal of Forecasting*, vol. 16, pp. 437–450, Oct. 2000, doi: 10.1016/S0169-2070(00)00065-0.

[AKK18] G. Atluri, A. Karpatne, and V. Kumar, "Spatio-Temporal Data Mining: A Survey of Problems and Methods," *ACM Comput. Surv.*, vol. 51, no. 4, pp. 83:1–83:41, 2018, doi: 10.1145/3161602.

[Yan+20] C. Yang, K. Clarke, S. Shekhar, and C. V. Tao, "Big Spatiotemporal Data Analytics: A Research and Innovation Frontier," *International Journal of Geographical Information Science*, vol. 34, no. 6, pp. 1075–1088, June 2020a, doi: 10.1080/13658816.2019.1698743.

[SP19] Z. Shi and L. S. C. Pun-Cheng, "Spatiotemporal Data Clustering: A Survey of Methods," *ISPRS International Journal of Geo-Information*, vol. 8, no. 3, p. 112, Mar. 2019, doi: 10.3390/ijgi8030112.

[Kis+] S. Kisilevich, F. Mansmann, M. Nanni, and S. Rinzivillo, "Spatio-Temporal Clustering: A Survey."

[Dua+07] L. Duan, L. Xu, F. Guo, J. Lee, and B. Yan, "A Local-Density Based Spatial Clustering Algorithm with Noise," *Information systems*, vol. 32, no. 7, pp. 978–986, 2007, doi: 10.1016/j.is.2006.10.006.

[LDR20] D. S. Lamb, J. Downs, and S. Reader, " Space-Time Hierarchical Clustering for Identifying Clusters in Spatiotemporal Point Data ," *ISPRS International Journal of Geo-Information*, vol. 9, no. 2, p. 85, Feb. 2020, doi: 10.3390/ijgi9020085.

[Hua+19] M. Huang, Q. Bao, Y. Zhang, and W. Feng, " A Hybrid Algorithm for Forecasting Financial Time Series Data Based on DBSCAN and SVR ," *Information*, vol. 10, no. 3, p. 103, Mar. 2019, doi: 10.3390/info10030103.

[Dor+24] O. Dorabiala, D. V. Dabke, J. Webster, N. Kutz, and A. Aravkin, "Spatiotemporal K-Means," no. arXiv:2211.05337. arXiv, Apr. 2024. doi: 10.48550/arXiv.2211.05337.

[KTC22] G. Klassen, M. Tatusch, and S. Conrad, "Cluster-Based Stability Evaluation in Time Series Data Sets," *Applied Intelligence (Dordrecht, Netherlands)*, pp. 1–24, Dec. 2022, doi: 10.1007/s10489-022-04231-7.

[Far+23] O. Faruque, F. N. Nji, M. Cham, R. M. Salvi, X. Zheng, and J. Wang, " Deep Spatiotemporal Clustering: A Temporal Clustering Approach for Multi-Dimensional Climate Data ," no. arXiv:2304.14541. arXiv, Sept. 2023. doi: 10.48550/arXiv.2304.14541.

[FPR00] A. Ferretti, C. Prati, and F. Rocca, " Nonlinear Subsidence Rate Estimation Using Permanent Scatterers in Differential SAR Interferometry ," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 38, no. 5, pp. 2202–2212, 2000, doi: 10.1109/36.868878.

[FPR01] A. Ferretti, C. Prati, and F. Rocca, "Permanent Scatterers in SAR Interferometry," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 39, no. 1, pp. 8–20, 2001, doi: 10.1109/36.898661.

[Gho+25] B. Ghoish, M. Motagh, M. Anvari, and S. Maghsudi, " Improving Atmospheric Noise Correction from InSAR Time Series Using Variational Autoencoder with Clustering (VAE-clustering) Method ," *Remote Sensing*, vol. 17, p. 3189, Sept. 2025, doi: 10.3390/rs17183189.

[Far+24] C. A. Farías, M. Lenardón Sánchez, R. Bonì, and F. Cigna, " Statistical and Independent Component Analysis of Sentinel-1 InSAR Time Series to Assess Land Subsidence Trends ," *Remote Sensing*, vol. 16, no. 21, p. 4066, Jan. 2024, doi: 10.3390/rs16214066.

[Xin+24] X. Xing *et al.*, " Independent Component Analysis (ICA) Based Method for Estimating the Deformation of Highways in Permafrost Region (HPICA)—a Case Study of Maduo Section of Gongyu Highway ," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 17, pp. 970–984, 2024, doi: 10.1109/JSTARS.2023.3336916.

[Ryg+23] M. Rygus, A. Novellino, E. Hussain, F. Syafiudin, H. Andreas, and C. Meisina, " A Clustering Approach for the Analysis of InSAR Time Series: Application to the Bandung Basin (Indonesia) ," *Remote Sensing*, vol. 15, no. 15, p. 3776, Jan. 2023, doi: 10.3390/rs15153776.

[Lac+] V. Lachi *et al.*, " Graph Neural Networks for Temporal Graphs: State of the Art, Open Challenges, and Opportunities ."

[ZYW24] Y. Zheng, L. Yi, and Z. Wei, "A Survey of Dynamic Graph Neural Networks," no. arXiv:2404.18211. arXiv, Apr. 2024. doi: 10.48550/arXiv.2404.18211.

[Li+23] Y. Li, Y. Shen, L. Chen, and M. Yuan, " Zebra: When Temporal Graph Neural Networks Meet Temporal Personalized PageRank ," *Proc. VLDB Endow.*, vol. 16, no. 6, pp. 1332–1345, Feb. 2023b, doi: 10.14778/3583140.3583150.

[Cou+21] C. Coupette, J. Beckedorf, D. Hartung, M. Bommarito, and D. M. Katz, " Measuring Law over Time: A Network Analytical Framework with an Application to Statutes and Regulations in the United States and Germany ," *Frontiers in Physics*, vol. 9, May 2021, doi: 10.3389/fphy.2021.658463.

[Yan+20] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. Salakhutdinov, and Q. V. Le, "XLNet: Generalized Autoregressive Pretraining for Language Understanding." [Online]. Available: https://arxiv.org/abs/1906.08237

[Kha+23] M. Khatri, M. Yusuf, Y. Kumar, R. R. Shah, and P. Kumaraguru, "Exploring Graph Neural Networks for Indian Legal Judgment Prediction," no. arXiv:2310.12800. arXiv, Oct. 2023. doi: 10.48550/arXiv.2310.12800.

[Zha+22] Q. Zhao, T. Gao, S. Zhou, D. Li, and Y. Wen, " Legal Judgment Prediction via Heterogeneous Graphs and Knowledge of Law Articles ," *Applied Sciences*, vol. 12, no. 5, p. 2531, Jan. 2022, doi: 10.3390/app12052531.

[AA24] E. K. Anastasiadis and I. Antoniou, "Directed Criminal Networks: Temporal Analysis and Disruption," *Information*, vol. 15, no. 2, p. 84, Feb. 2024, doi: 10.3390/info15020084.

[Lia+25] Y. Liang *et al.*, " Foundation Models for Spatio-Temporal Data Science: A Tutorial and Survey ," no. arXiv:2503.13502. arXiv, Mar. 2025. doi: 10.48550/arXiv.2503.13502.

[Goo+25] A. Goodge, W. S. Ng, B. Hooi, and S. K. Ng, "Spatio-Temporal Foundation Models: Vision, Challenges, and Opportunities," no. arXiv:2501.09045. arXiv, Feb. 2025. doi: 10.48550/arXiv.2501.09045.

[Zho+25] S. Zhong *et al.*, "Learning to Factorize Spatio-Temporal Foundation Models," in *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, Oct. 2025b.

[Yua+25] Y. Yuan, J. Ding, C. Han, Z. Sheng, D. Jin, and Y. Li, " UniFlow: A Foundation Model for Unified Urban Spatio-Temporal Flow Prediction ," no. arXiv:2411.12972. arXiv, Apr. 2025. doi: 10.48550/arXiv.2411.12972.

[Che+25] S. Chen, G. Long, J. Jiang, and C. Zhang, "Federated Foundation Models on Heterogeneous Time Series," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 39, no. 15, pp. 15839–15847, Apr. 2025, doi: 10.1609/aaai.v39i15.33739.

[AKB23] A. Afroozeh, L. X. Kuffo, and P. Boncz, "ALP: Adaptive Lossless Floating-Point Compression," *Proc. ACM Manag. DATA*, vol. 1, no. 4, pp. 230:1–230:26, 2023, doi: 10.1145/3626717.

[MHM25] J. G. Matt, P. Huang, and B. Maag, " Lossless Compression of Time Series Data: A Comparative Study ," no. arXiv:2510.07015. arXiv, Oct. 2025. doi: 10.48550/arXiv.2510.07015.

[Deu96] L. P. Deutsch, "GZIP File Format Specification Version 4.3," Request for Comments RFC1952, May 1996. doi: 10.17487/RFC1952.

[Sna] "Snappy."

[Zen+23] X. Zeng, Y. Hui, J. Shen, A. Pavlo, W. McKinney, and H. Zhang, "An Empirical Evaluation of Columnar Storage Formats," *Proc. VLDB Endow.*, vol. 17, no. 2, pp. 148–161, 2023, doi: 10.14778/3626292.3626298.

[HPZ11] C. Hoobin, S. J. Puglisi, and J. Zobel, " Relative Lempel-Ziv Factorization for Efficient Storage and Retrieval of Web Collections ," no. arXiv:1106.2587. arXiv, Dec. 2011. doi: 10.48550/arXiv.1106.2587.

[Tan+25] Y. Tang *et al.*, "Improving Time Series Data Compression in Apache IoTDB," *Proc. VLDB Endow.*, vol. 18, no. 10, pp. 3406–3420, 2025, doi: 10.14778/3748191.3748204.

[Sah+] A. Saha, A. Raman, R. Marcus, and M. Athanassoulis, " Exploring Wavelet Trees as Space-Efficient Physical-to-Sorted Mapping for Learned Indexes ."

[Bab+15] M. Babenko, P. Gawrychowski, T. Kociumaka, and T. Starikovskaya, "Wavelet Trees Meet Suffix Trees," in *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms*, Oct. 2015, pp. 572–591. doi: 10.1137/1.9781611973730.39.

[Shu20] J. Shun, " Improved Parallel Construction of Wavelet Trees and Rank/Select Structures ," *Information and Computation*, vol. 273, p. 104516, Aug. 2020, doi: 10.1016/j.ic.2020.104516.

[Bri+09] N. R. Brisaboa, M. R. Luaces, G. Navarro, and D. Seco, "A New Point Access Method Based on Wavelet Trees," in *Advances in Conceptual Modeling - Challenging Perspectives*, C. A. Heuser and G. Pernul, Eds., Berlin, Heidelberg: Springer, 2009, pp. 297–306. doi: 10.1007/978-3-642-04947-7_36.

[KP] J. H. Knudsen and R. L. Pedersen, "Engineering Rank and Select Queries on Wavelet Trees."

[CN12] F. Claude and G. Navarro, "The Wavelet Matrix," in *String Processing and Information Retrieval*, L. Calderón-Benavides, C. González-Caro, E. Chávez, and N. Ziviani, Eds., Berlin, Heidelberg: Springer, 2012, pp. 167–179. doi: 10.1007/978-3-642-34109-0_18.

[Pel+15] T. Pelkonen *et al.*, "Gorilla: a fast, scalable, in-memory time series database," *Proc. VLDB Endow.*, vol. 8, no. 12, pp. 1816–1827, Aug. 2015, doi: 10.14778/2824032.2824078.

[LZ77] A. Lempel and J. Ziv, "A universal algorithm for sequential data compression," *IEEE Transactions on Information Theory*, vol. 23, no. 3, pp. 337–343, 1977, doi: 10.1109/TIT.1977.1055714.

[MBP25] C. E. Muñiz-Cuza, M. Boehm, and T. B. Pedersen, " CAMEO: Autocorrelation-Preserving Line Simplification for Lossy Time Series Compression ," no. arXiv:2501.14432. arXiv, Jan. 2025. doi: 10.48550/arXiv.2501.14432.

[Pro26] "Prometheus/Prometheus." Jan. 2026.

[Inf] "InfluxData."

[Tim26] "Timescale/Timescaledb." Jan. 2026.

[Li+23] R. Li *et al.*, " Erasing-Based Lossless Compression Method for Streaming Floating-Point Time Series ," no. arXiv:2306.16053. arXiv, June 2023c. doi: 10.48550/arXiv.2306.16053.

[BMG18] D. Blalock, S. Madden, and J. Guttag, "Sprintz: Time Series Compression for the Internet of Things," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 2, no. 3, pp. 1–23, Sept. 2018, doi: 10.1145/3264903.

[Lia+24] P. Liakos, K. Papakonstantinopoulou, T. Bruineman, M. Raasveldt, and Y. Kotidis, " How to Make your Duck Fly: Advanced Floating Point Compression to the Rescue ," in *EDBT*, 2024, pp. 826–829. [Online]. Available: https://doi.org/10.48786/edbt.2024.80

[Li+23] R. Li, Z. Li, Y. Wu, C. Chen, and Y. Zheng, "Elf: Erasing-Based Lossless Floating-Point Compression," *Proc. VLDB Endow.*, vol. 16, no. 7, pp. 1763–1776, Mar. 2023d, doi: 10.14778/3587136.3587149.

[Sen+18] P. Senin *et al.*, " GrammarViz 3.0: Interactive Discovery of Variable-Length Time Series Patterns ," *ACM Trans. Knowl. Discov. DATA*, vol. 12, no. 1, pp. 10:1–10:28, Feb. 2018, doi: 10.1145/3051126.

[Lin+07] J. Lin, E. Keogh, L. Wei, and S. Lonardi, "Experiencing SAX: A Novel Symbolic Representation of Time Series," *Data Mining and Knowledge Discovery*, vol. 15, no. 2, pp. 107–144, Oct. 2007, doi: 10.1007/s10618-007-0064-z.

[GLR17] Y. Gao, J. Lin, and H. Rangwala, " IterativE Grammar-Based Framework for Discovering Variable-Length Time Series Motifs ," in *2017 IEEE International Conference on Data Mining (ICDM)*, Nov. 2017, pp. 111–116. doi: 10.1109/ICDM.2017.20.

[HD25] P. Heringer and D. Doerr, "Human Readable Compression of GFA Paths Using Grammar-Based Code." bioRxiv, p. 2025.05.22.655470, May 2025. doi: 10.1101/2025.05.22.655470.

[GG10] T. Gagie and P. Gawrychowski, "Grammar-Based Compression in a Streaming Model," no. arXiv:912.0850. arXiv, Feb. 2010. doi: 10.48550/arXiv.0912.0850.

[Sen+] P. Senin, J. Lin, X. Wang, T. Oates, and S. Gandhi, "Time Series Anomaly Discovery with Grammar-Based Compression."

[KN21] A. Kathpalia and N. Nagaraj, "Time-Reversibility, Causality and Compression-Complexity," *Entropy*, vol. 23, no. 3, p. 327, Mar. 2021, doi: 10.3390/e23030327.

[B⁄P17] P. Bille, I. L. ⁄rtz, and N. Prezza, "Practical and Effective Re-Pair Compression." [Online]. Available: https://arxiv.org/abs/1704.08558

[Kim+24] J. Kim, R. Varki, M. Oliva, and C. Boucher, " Re2Pair: Increasing the Scalability of RePair by Decreasing Memory Usage ," *bioRxiv*, p. 2024.07.11.603142, July 2024, doi: 10.1101/2024.07.11.603142.

[KPZ10] S. Kuruppu, S. J. Puglisi, and J. Zobel, " Relative Lempel-Ziv Compression of Genomes for Large-Scale Storage and Retrieval ," in *String Processing and Informa-*

tion Retrieval, E. Chavez and S. Lonardi, Eds., Berlin, Heidelberg: Springer, 2010, pp. 201–206. doi: 10.1007/978-3-642-16321-0_20.

[PZ20] S. J. Puglisi and B. Zhukova, "Relative Lempel-Ziv Compression of Suffix Arrays," in *String Processing and Information Retrieval*, C. Boucher and S. V. Thankachan, Eds., Cham: Springer International Publishing, 2020, pp. 89–96. doi: 10.1007/978-3-030-59212-7_7.

[Bil+22] P. Bille, I. L. Gørtz, S. J. Puglisi, and S. R. Tarnow, "Hierarchical Relative Lempel-Ziv Compression," no. arXiv:2208.11371. arXiv, Aug. 2022. doi: 10.48550/arXiv.2208.11371.

[KPZ] S. Kuruppu, S. J. Puglisi, and J. Zobel, "Optimized Relative Lempel-Ziv Compression of Genomes."

[NS25] S. R. Nelluri and F. A. A. Saldanha, " Mastering Big Data Formats: ORC, Parquet, Avro, Iceberg, and the Strategy of Selection ," *International Journal of Computer Trends and Technology - IJCTT*, vol. 73, no. 1, pp. 44–50, Jan. 2025, doi: 10.14445/22312803/IJCTT-V73I1P105.

[Liu+24] C. Liu, A. Pavlenko, M. Interlandi, and B. Haynes, " Data Formats in Analytical DBMSs: Performance Trade-Offs and Future Directions ," no. arXiv:2411.14331. arXiv, Nov. 2024a. doi: 10.48550/arXiv.2411.14331.

[Wan+24] W. Wan *et al.*, "Compressed Data Direct Computing for Databases," *IEEE Transactions on Knowledge and Data Engineering*, vol. 36, no. 5, pp. 1902–1918, May 2024a, doi: 10.1109/TKDE.2023.3316274.

[Mos+25] M. H. Moslemi, A. Mousavi, B. Behkamal, and M. Milani, "Heterogeneity in Entity Matching: A Survey and Experimental Analysis," no. arXiv:2508.08076. arXiv, Aug. 2025. doi: 10.48550/arXiv.2508.08076.

[Kon+16] P. Konda *et al.*, " Magellan: Toward Building Entity Matching Management Systems over Data Science Stacks ," *Proceedings of the VLDB Endowment*, vol. 9, pp. 1581–1584, Sept. 2016, doi: 10.14778/3007263.3007314.

[Dev+19] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, " BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding ." [Online]. Available: https://arxiv.org/abs/1810.04805

[Liu+19] Y. Liu *et al.*, "RoBERTa: A Robustly Optimized BERT Pretraining Approach." [Online]. Available: https://arxiv.org/abs/1907.11692

[ZS24] H. Zhang and M. O. Shafiq, " Survey of Transformers and towards Ensemble Learning Using Transformers for Natural Language Processing ," *Journal of Big DATA*, vol. 11, no. 1, p. 25, 2024, doi: 10.1186/s40537-023-00842-0.

[Ope+24] OpenAI *et al.*, "GPT-4 Technical Report." [Online]. Available: https://arxiv.org/abs/2303.08774

[Tou+23] H. Touvron *et al.*, "LLaMA: Open and Efficient Foundation Language Models." [Online]. Available: https://arxiv.org/abs/2302.13971

[Bal+25] R. I. A. Baldivas *et al.*, " LegalEye: Multimodal Court Deception Detection across Multiple Languages ," *Behavioral Sciences*, vol. 15, no. 12, p. 1707, Dec. 2025, doi: 10.3390/bs15121707.

[NSK25] F. Naudot, T. Sundqvist, and T. Kampik, "llmSHAP: A Principled Approach to LLM Explainability," no. arXiv:2511.01311. arXiv, Nov. 2025. doi: 10.48550/arXiv.2511.01311.

[RSG16] M. T. Ribeiro, S. Singh, and C. Guestrin, ""Why Should I Trust You?": Explaining the Predictions of Any Classifier." [Online]. Available: https://arxiv.org/abs/1602.04938

[Wad+24] S. Wadhwa, A. Krishnan, R. Wang, B. C. Wallace, and L. Kong, " Learning from Natural Language Explanations for Generalizable Entity Matching ," in *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, Y. Al-Onaizan, M. Bansal, and Y.-N. Chen, Eds., Miami, Florida, USA: Association for Computational Linguistics, Nov. 2024, pp. 6114–6129. doi: 10.18653/v1/2024.emnlp-main.352.

[Lew+21] P. Lewis *et al.*, "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks." [Online]. Available: https://arxiv.org/abs/2005.11401

[Nae+24] Z. A. Naeem, M. S. Ahmad, M. Eltabakh, M. Ouzzani, and N. Tang, " RetClean: Retrieval-Based Data Cleaning Using Foundation Models and Data Lakes ," no. arXiv:2303.16909. arXiv, Dec. 2024. doi: 10.48550/arXiv.2303.16909.

[PDB23] R. Peeters, R. C. Der, and C. Bizer, "WDC Products: A Multi-Dimensional Entity Matching Benchmark," no. arXiv:2301.09521. arXiv, June 2023. doi: 10.48550/arXiv.2301.09521.

[Xie+23] T. Xie, K. Dai, K. Wang, R. Li, and L. Zhao, " DeepMatcher: A Deep Transformer-based Network for Robust and Accurate Local Feature Matching ." [Online]. Available: https://arxiv.org/abs/2301.02993

[Li+20] Y. Li, J. Li, Y. Suhara, A. Doan, and W.-C. Tan, "Deep Entity Matching with Pre-Trained Language Models," *Proceedings of the VLDB Endowment*, vol. 14, no. 1, pp. 50–60, Sept. 2020a, doi: 10.14778/3421424.3421431.

[PB22] R. Peeters and C. Bizer, "Supervised Contrastive Learning for Product Matching," in *Companion Proceedings of the Web Conference 2022*, in WWW '22. ACM, Apr. 2022, pp. 248–251. doi: 10.1145/3487553.3524254.

[Pag+22] M. Paganelli, F. D. Buono, A. Baraldi, and F. Guerra, "Analyzing How BERT Performs Entity Matching," *Proceedings of the VLDB Endowment*, vol. 15, no. 8, pp. 1726–1738, Apr. 2022, doi: 10.14778/3529337.3529356.

[Par+25] F. D. M. Pardo *et al.*, " GraLMatch: Matching Groups of Entities with Graphs and Language Models ." OpenProceedings.org, 2025. doi: 10.48786/EDBT.2025.01.

[Jia+24] A. Q. Jiang *et al.*, "Mixtral of Experts." [Online]. Available: https://arxiv.org/abs/2401.04088

[PSB24] R. Peeters, A. Steiner, and C. Bizer, "Entity Matching Using Large Language Models," no. arXiv:2310.11244. arXiv, Oct. 2024. doi: 10.48550/arXiv.2310.11244.

[San+20] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, " DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter ." [Online]. Available: https://arxiv.org/abs/1910.01108

[Zha+25] Z. Zhang, P. Groth, I. Calixto, and S. Schelter, " A Deep Dive into Cross-Dataset Entity Matching with Large and Small Language Models. ," in *EDBT*, A. Simitsis, B. Kemme, A. Queralt, O. Romero, and P. Jovanovic, Eds., OpenProceedings.org, 2025, pp. 922–934. doi: 10.48786/edbt.2025.75.

[Tu+23] J. Tu *et al.*, " Unicorn: A Unified Multi-tasking Model for Supporting Matching Tasks in Data Integration ," *Proc. ACM Manag. Data*, vol. 1, no. 1, May 2023, doi: 10.1145/3588938.

[RFS25] Y. Rahulamathavan, M. Farooq, and V. D. Silva, " PLEX: Perturbation-Free Local Explanations for LLM-based Text Classification ," no. arXiv:2507.10596. arXiv, July 2025. doi: 10.48550/arXiv.2507.10596.

[Sal+25] A. M. Salih *et al.*, " A Perspective on Explainable Artificial Intelligence Methods: SHAP and LIME ," *Advanced Intelligent Systems*, vol. 7, no. 1, p. 2400304, 2025, doi: 10.1002/aisy.202400304.

[Yan+25] A. Yang *et al.*, "Qwen3 Technical Report." [Online]. Available: https://arxiv.org/abs/2505.09388

[Li+25] Z. Li, X. Wu, H. Du, F. Liu, H. Nghiem, and G. Shi, " A Survey of State of the Art Large Vision Language Models: Alignment, Benchmark, Evaluations and Challenges ," no. arXiv:2501.02189. arXiv, Apr. 2025a. doi: 10.48550/arXiv.2501.02189.

[Sim24] J. Siml, " Beyond Black Box: Enhancing Model Explainability with LLMs and SHAP ." Jan. 2024.

[Bab24] D. Babu, "Using LLMs for SHAP Explanation." Apr. 2024.

[Liu+25] J. Liu *et al.*, "Legal Fact Prediction: The Missing Piece in Legal Judgment Prediction," no. arXiv:2409.07055. arXiv, Nov. 2025b. doi: 10.48550/arXiv.2409.07055.

[Lin+25] J. Lin, S. Wang, X. Guo, J. Shun, and Y. Zhu, " Temporal Reasoning with Large Language Models Augmented by Evolving Knowledge Graphs ," no. arXiv:2509.15464. arXiv, Sept. 2025. doi: 10.48550/arXiv.2509.15464.

[JC22] E. Jacob de Menezes-Neto and M. B. M. Clementino, " Using Deep Learning to Predict Outcomes of Legal Appeals Better than Human Experts: A Study with Data from Brazilian Federal Courts ," *PLOS One*, vol. 17, no. 7, p. e272287, July 2022, doi: 10.1371/journal.pone.0272287.

[Li+25] A. Li *et al.*, " Legal Judgment Prediction Based on Knowledge-Enhanced Multi-Task and Multi-Label Text Classification ," in *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, L. Chiruzzo, A. Ritter, and L. Wang, Eds., Albuquerque, New Mexico: Association for Computational Linguistics, Apr. 2025b, pp. 6957–6970. doi: 10.18653/v1/2025.naacl-long.355.

[FBS25] R. Friel, M. Belyi, and A. Sanyal, " RAGBench: Explainable Benchmark for Retrieval-Augmented Generation Systems ," no. arXiv:2407.11005. arXiv, Jan. 2025. doi: 10.48550/arXiv.2407.11005.

[Mar25] H. de Martim, " Graph RAG for Legal Norms: A Hierarchical, Temporal and Deterministic Approach ," no. arXiv:2505.00039. arXiv, June 2025. doi: 10.48550/arXiv.2505.00039.

[GPM25] F. Granata, F. Poggi, and M. Mongiovì, " Enhancing Retrieval-Augmented Generation with Entity Linking for Educational Platforms ," no. arXiv:2512.05967. arXiv, Dec. 2025. doi: 10.48550/arXiv.2512.05967.

[Che+21] M. Chen *et al.*, "Evaluating Large Language Models Trained on Code." [Online]. Available: https://arxiv.org/abs/2107.03374

[Yan+25] Z. Yang, T. Peng, C. Gao, C. Wang, H. Huang, and Y. Deng, " A Deep Dive into Retrieval-Augmented Generation for Code Completion: Experience on WeChat ," no. arXiv:2507.18515. arXiv, July 2025b. doi: 10.48550/arXiv.2507.18515.

[Jin+24] L. Jing *et al.*, " DSBench: How Far Are Data Science Agents from Becoming Data Science Experts? ," in *The Thirteenth International Conference on Learning Representations*, Oct. 2024.

[Ouy+25] S. Ouyang, D. Huang, J. Guo, Z. Sun, Q. Zhu, and J. M. Zhang, " DSCodeBench: A Realistic Benchmark for Data Science Code Generation ," no. arXiv:2505.15621. arXiv, Nov. 2025. doi: 10.48550/arXiv.2505.15621.

[Orb19] Orbifold, "The Cora Dataset." Sept. 2019.

[KTR10] H. Köpcke, A. Thor, and E. Rahm, "Evaluation of entity resolution approaches on real-world match problems," *Proc. VLDB Endow.*, vol. 3, no. 1–2, pp. 484–493, Sept. 2010, doi: 10.14778/1920841.1920904.

[Wu+24] J. Wu *et al.*, " HierGAT: Hierarchical Spatial-Temporal Network with Graph and Transformer for Video HOI Detection ," *Multimedia Systems*, vol. 31, no. 1, p. 13, Dec. 2024, doi: 10.1007/s00530-024-01604-5.

[Pot25] B. Potts, "BenchmarkQED: Automated Benchmarking of RAG Systems." June 2025.

[KNV23] C. Kosma, G. Nikolentzos, and M. Vazirgiannis, " Time-Parameterized Convolutional Neural Networks for Irregularly Sampled Time Series ," no. arXiv:2308.03210. arXiv, Aug. 2023. doi: 10.48550/arXiv.2308.03210.

[Oh+25] Y. Oh, S. Kam, J. Lee, D.-Y. Lim, S. Kim, and A. Bui, " Comprehensive Review of Neural Differential Equations for Time Series Analysis ," no. arXiv:2502.09885. arXiv, Sept. 2025. doi: 10.48550/arXiv.2502.09885.

[Cha+23] Z. Chang, S. Liu, R. Qiu, S. Song, Z. Cai, and G. Tu, " Time-Aware Neural Ordinary Differential Equations for Incomplete Time Series Modeling ," *Journal of Supercomputing*, pp. 1–29, May 2023, doi: 10.1007/s11227-023-05327-8.

[Pat+24] A. Patharkar, F. Cai, F. Al-Hindawi, and T. Wu, " Predictive Modeling of Biomedical Temporal Data in Healthcare Applications: Review and Future Directions ," *Frontiers in Physiology*, vol. 15, Oct. 2024, doi: 10.3389/fphys.2024.1386760.

[GLZ24] G. O. Ghosheh, J. Li, and T. Zhu, " Understanding Missingness in Time-series Electronic Health Records for Individualized Representation ," no. arXiv:2402.15730. arXiv, Feb. 2024. doi: 10.48550/arXiv.2402.15730.

[Kid+20] P. Kidger, J. Morrill, J. Foster, and T. Lyons, "Neural Controlled Differential Equations for Irregular Time Series," in *Advances in Neural Information Processing Systems*, Curran Associates, Inc., 2020, pp. 6696–6707.

[LLY23] Z. Li, S. Li, and X. Yan, " Time Series as Images: Vision Transformer for Irregularly Sampled Time Series ," *Advances in Neural Information Processing Systems*, vol. 36, pp. 49187–49204, Dec. 2023.

[RCD19] Y. Rubanova, R. T. Q. Chen, and D. K. Duvenaud, " Latent Ordinary Differential Equations for Irregularly-Sampled Time Series ," in *Advances in Neural Information Processing Systems*, Curran Associates, Inc., 2019.

[Li+20] X. Li, T.-K. L. Wong, R. T. Q. Chen, and D. Duvenaud, "Scalable Gradients for Stochastic Differential Equations," in *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, PMLR, June 2020b, pp. 3870–3882.

[Zha+24] K. Zhang *et al.*, " Self-Supervised Learning for Time Series Analysis: Taxonomy, Progress, and Prospects ," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 46, no. 10, pp. 6775–6794, Oct. 2024a, doi: 10.1109/TPAMI.2024.3387317.

[Tan+20] X. Tang, H. Yao, Y. Sun, C. Aggarwal, P. Mitra, and S. Wang, " Joint Modeling of Local and Global Temporal Dynamics for Multivariate Time Series Forecasting with Missing Values ," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 4, pp. 5956–5963, Apr. 2020, doi: 10.1609/aaai.v34i04.6056.

[Ran25] C. Rand, "Optimizing Transformer Models for Variable-Length Input Sequences." July 2025.

[Feg+22] P. Fegade, T. Chen, P. Gibbons, and T. Mowry, " The CoRa Tensor Compiler: Compilation for Ragged Tensors with Minimal Padding ," *Proceedings of Machine Learning and Systems*, vol. 4, pp. 721–747, Apr. 2022.

[Hor+20] M. Horn, M. Moor, C. Bock, B. Rieck, and K. Borgwardt, "Set Functions for Time Series," no. arXiv:1909.12064. arXiv, Sept. 2020. doi: 10.48550/arXiv.1909.12064.

[LAC17] M. Lepot, J.-B. Aubin, and F. H. L. R. Clemens, " Interpolation in Time Series: An Introductive Overview of Existing Methods, Their Performance Criteria and Uncertainty Assessment ," *Water*, vol. 9, no. 10, p. 796, Oct. 2017, doi: 10.3390/w9100796.

[Per+22] A. Perez-Lebel, G. Varoquaux, M. Le~Morvan, J. Josse, and J.-B. Poline, " Benchmarking Missing-Values Approaches for Predictive Models on Health Databases ," *GigaScience*, vol. 11, p. giac13, Jan. 2022, doi: 10.1093/gigascience/giac013.

[SM19] S. N. Shukla and B. M. Marlin, "Interpolation-Prediction Networks for Irregularly Sampled Time Series," no. arXiv:1909.07782. arXiv, Sept. 2019. doi: 10.48550/arXiv.1909.07782.

[Neo+25] A. Neog *et al.*, " Investigating a Model-Agnostic and Imputation-Free Approach for Irregularly-Sampled Multivariate Time-Series Modeling ," no. arXiv:2502.15785. arXiv, Sept. 2025. doi: 10.48550/arXiv.2502.15785.

[Liu+25] X. Liu *et al.*, " Rethinking Irregular Time Series Forecasting: A Simple yet Effective Baseline ," no. arXiv:2505.11250. arXiv, May 2025c. doi: 10.48550/arXiv.2505.11250.

[RUB76] D. B. RUBIN, "Inference and missing data," *Biometrika*, vol. 63, no. 3, pp. 581–592, 1976, doi: 10.1093/biomet/63.3.581.

[Emm+21] T. Emmanuel, T. Maupong, D. Mpoeleng, T. Semong, B. Mphago, and O. Tabona, "A Survey on Missing Data in Machine Learning," *Journal of Big Data*, vol. 8, no. 1, p. 140, 2021, doi: 10.1186/s40537-021-00516-9.

[ZAB24] Y. Zhou, S. Aryal, and M. R. Bouadjenek, "Review for Handling Missing Data with Special Missing Mechanism," no. arXiv:2404.04905. arXiv, Apr. 2024. doi: 10.48550/arXiv.2404.04905.

[LKW] Z. C. Lipton, D. C. Kale, and R. Wetzel, "Modeling Missing Data in Clinical Time Series with RNNs."

[Gro20] R. H. H. Groenwold, " Informative Missingness in Electronic Health Record Systems: The Curse of Knowing ," *Diagnostic and Prognostic Research*, vol. 4, p. 8, July 2020, doi: 10.1186/s41512-020-00077-0.

[SSO21] J. Singh, M. Sato, and T. Ohkuma, " On Missingness Features in Machine Learning Models for Critical Care: Observational Study ," *JMIR Medical Informatics*, vol. 9, no. 12, p. e25022, Dec. 2021, doi: 10.2196/25022.

[Wan+24] J. Wang *et al.*, "Deep Learning for Multivariate Time Series Imputation: A Survey," no. arXiv:2402.04059. arXiv, Feb. 2024b. doi: 10.48550/arXiv.2402.04059.

[Che+16] Z. Che, S. Purushotham, K. Cho, D. Sontag, and Y. Liu, " Recurrent Neural Networks for Multivariate Time Series with Missing Values ." [Online]. Available: https://arxiv.org/abs/1606.01865

[NPL16] D. Neil, M. Pfeiffer, and S.-C. Liu, " Phased LSTM: Accelerating Recurrent Network Training for Long or Event-based Sequences ." [Online]. Available: https://arxiv.org/abs/1610.09513

[HP20] M. Habiba and B. A. Pearlmutter, "Neural ODEs for Informative Missingness in Multivariate Time Series," no. arXiv:2005.10693. arXiv, May 2020. doi: 10.48550/arXiv.2005.10693.

[Jhi+23] S. Y. Jhin, M. Jo, S. Kook, and N. Park, "Learnable Path in Neural Controlled Differential Equations," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, no. 7, pp. 8014–8022, June 2023, doi: 10.1609/aaai.v37i7.25969.

[FGS25] Y. Fang, Q. L. Gia, and F. Salim, " SDE-attention: Latent Attention in SDE-RNNs for Irregularly Sampled Time Series with Missing Data ," no. arXiv:2511.23238. arXiv, Nov. 2025. doi: 10.48550/arXiv.2511.23238.

[Li24] X. Li, " Time Series Forecasting with Missing Data Using Generative Adversarial Networks and Bayesian Inference ," *Information*, vol. 15, no. 4, p. 222, Apr. 2024, doi: 10.3390/info15040222.

[ZGK23] S. Zeng, F. Graf, and R. Kwitt, "Latent SDEs on Homogeneous Spaces," *Advances in Neural Information Processing Systems*, vol. 36, pp. 76151–76180, Dec. 2023.

[Den+21] R. Deng, M. A. Brubaker, G. Mori, and A. Lehrmann, "Continuous Latent Process Flows," in *Advances in Neural Information Processing Systems*, Curran Associates, Inc., 2021, pp. 5162–5173.

[Joh+16] A. E. Johnson *et al.*, "MIMIC-III, a Freely Accessible Critical Care Database," *Scientific Data*, vol. 3, no. 1, p. 160035, May 2016, doi: 10.1038/sdata.2016.35.

[Har+19] H. Harutyunyan, H. Khachatrian, D. C. Kale, G. Ver Steeg, and A. Galstyan, "Multitask Learning and Benchmarking with Clinical Time Series Data," *Scientific Data*, vol. 6, p. 96, June 2019, doi: 10.1038/s41597-019-0103-9.

[RBH22] E. Röösli, S. Bozkurt, and T. Hernandez-Boussard, " Peeking into a Black Box, the Fairness and Generalizability of a MIMIC-III Benchmarking Model ," *Scientific Data*, vol. 9, p. 24, Jan. 2022, doi: 10.1038/s41597-021-01110-7.

[Zan25] M. Zanotti, "On the Retraining Frequency of Global Forecasting Models," no. arXiv:2505.00356. arXiv, July 2025. doi: 10.48550/arXiv.2505.00356.

[Klö+25] C. Klötergens, V. K. Yalavarthi, R. Scholz, M. Stubbemann, S. Born, and L. Schmidt-Thieme, " Physiome-ODE: A Benchmark for Irregularly Sampled Multivariate Time Series Forecasting Based on Biological ODEs ," no. arXiv:2502.07489. arXiv, Feb. 2025. doi: 10.48550/arXiv.2502.07489.

[DHS22] D. Dobrev, D. Hansen, and P. Szerszen, "A Randomized Missing Data Approach to Robust Filtering and Forecasting," no. arXiv:2104.14664. arXiv, Oct. 2022. doi: 10.48550/arXiv.2104.14664.

[DS25] D. Dobrev and P. J. Szerszeń, " Missing Data Substitution for Enhanced Robust Filtering and Forecasting in Linear State-Space Models ," *Finance and Economics Discussion Series*, no. 2025–1, pp. 1–38, Jan. 2025, doi: 10.17016/feds.2025.001.

[TCK25] A. A. Toye, A. Celik, and S. Kleinberg, " Benchmarking Missing Data Imputation Methods for Time Series Using Real-World Test Cases ," *Proceedings of Machine Learning Research*, vol. 287, pp. 480–501, June 2025.

[Lu+25] Z. Lu *et al.*, " Benchmarking Probabilistic Time Series Forecasting Models on Neural Activity ," no. arXiv:2510.18037. arXiv, Oct. 2025. doi: 10.48550/arXiv.2510.18037.

[KM25] H. Katz and T. Maierhofer, " Forecasting the U.S. Renewable-Energy Mix with an ALR-BDARMA Compositional Time-Series Framework ," *Forecasting*, vol. 7, no. 4, p. 62, Dec. 2025, doi: 10.3390/forecast7040062.

[WJG24] B. Wang, J. Jennings, and W. Gong, "Neural Structure Learning with Stochastic Differential Equations," no. arXiv:2311.03309. arXiv, May 2024. doi: 10.48550/arXiv.2311.03309.

[MPW25] F. Montet, B. Pasquier, and B. Wolf, " Benchmarking Foundation Models for Time-Series Forecasting: Zero-Shot, Few-Shot, and Full-Shot Evaluations ," *Computer Sciences & Mathematics Forum*, vol. 11, no. 1, p. 32, 2025, doi: 10.3390/cmsf2025011032.

[Du+25] W. Du, Y. Yang, L. Qian, J. Wang, and Q. Wen, " PyPOTS: A Python Toolkit for Machine Learning on Partially-Observed Time Series ." [Online]. Available: https://arxiv.org/abs/2305.18811

[Du+24] W. Du *et al.*, "TSI-Bench: Benchmarking Time Series Imputation." [Online]. Available: https://arxiv.org/abs/2406.12747

[RAP] "RAPIDS | GPU Accelerated Data Science."

[WSP22] P. Wenig, S. Schmidl, and T. Papenbrock, " TimeEval: a benchmarking toolkit for time series anomaly detection algorithms ," *Proc. VLDB Endow.*, vol. 15, no. 12, pp. 3678–3681, Aug. 2022, doi: 10.14778/3554821.3554873.

[LJD19] S. Liu, E. Johns, and A. J. Davison, "End-to-End Multi-Task Learning with Attention." [Online]. Available: https://arxiv.org/abs/1803.10704

[Cho+23] R. R. Chowdhury, J. Li, X. Zhang, D. Hong, R. K. Gupta, and J. Shang, "PrimeNet: Pre-training for Irregular Multivariate Time Series," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, no. 6, pp. 7184–7192, June 2023, doi: 10.1609/aaai.v37i6.25876.

[VKB21] C. B. Vennerød, A. Kjærran, and E. S. Bugge, "Long Short-term Memory RNN." [Online]. Available: https://arxiv.org/abs/2105.06756

[Lea+16] C. Lea, M. D. Flynn, R. Vidal, A. Reiter, and G. D. Hager, "Temporal Convolutional Networks for Action Segmentation and Detection." [Online]. Available: https://arxiv.org/abs/1611.05267

[Nie+23] Y. Nie, N. H. Nguyen, P. Sinthong, and J. Kalagnanam, "A Time Series is Worth 64 Words: Long-term Forecasting with Transformers." [Online]. Available: https://arxiv.org/abs/2211.14730

[Lim+20] B. Lim, S. O. Arik, N. Loeff, and T. Pfister, " Temporal Fusion Transformers for Interpretable Multi-horizon Time Series Forecasting ." [Online]. Available: https://arxiv.org/abs/1912.09363

[Cha+22] C. Challu, K. G. Olivares, B. N. Oreshkin, F. Garza, M. Mergenthaler-Canseco, and A. Dubrawski, "N-HiTS: Neural Hierarchical Interpolation for Time Series Forecasting." [Online]. Available: https://arxiv.org/abs/2201.12886

[Cao+21] D. Cao *et al.*, " Spectral Temporal Graph Neural Network for Multivariate Time-series Forecasting ." [Online]. Available: https://arxiv.org/abs/2103.07719

[Nie+23] Y. Nie, N. H. Nguyen, P. Sinthong, and J. Kalagnanam, "A Time Series Is Worth 64 Words: Long-Term Forecasting with Transformers," no. arXiv:2211.14730. arXiv, Mar. 2023b. doi: 10.48550/arXiv.2211.14730.

[Her+22] J. Herzen *et al.*, "Darts: User-Friendly Modern Machine Learning for Time Series." [Online]. Available: https://arxiv.org/abs/2110.03224

[CG16] T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting System," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, in KDD '16. ACM, Aug. 2016, pp. 785–794. doi: 10.1145/2939672.2939785.

[SKB24] S. Sim, D. Kim, and H. Bae, " Correlation recurrent units: A novel neural architecture for improving the predictive performance of time-series data ." [Online]. Available: https://arxiv.org/abs/2211.16653

[Zha+24] Y. Zhang, R. Wu, S. M. Dascalu, and F. C. Harris, " A Novel Extreme Adaptive GRU for Multivariate Time Series Forecasting ," *Scientific Reports*, vol. 14, no. 1, p. 2991, Feb. 2024b, doi: 10.1038/s41598-024-53460-y.

[DCL23] W. Du, D. Côté, and Y. Liu, "SAITS: Self-attention-based imputation for time series," *Expert Systems with Applications*, vol. 219, p. 119619, June 2023, doi: 10.1016/j.eswa.2023.119619.

[Cao+18] W. Cao, D. Wang, J. Li, H. Zhou, L. Li, and Y. Li, "BRITS: Bidirectional Recurrent Imputation for Time Series." [Online]. Available: https://arxiv.org/abs/1805.10572

[You+25] L. You, J. Lu, X. Huang, and X. Nie, "FRET: Feature Redundancy Elimination for Test Time Adaptation." [Online]. Available: https://arxiv.org/abs/2505.10641

[Tas+21] Y. Tashiro, J. Song, Y. Song, and S. Ermon, " CSDI: Conditional Score-based Diffusion Models for Probabilistic Time Series Imputation ." [Online]. Available: https://arxiv.org/abs/2107.03502

[AK23] A. Akram and N. Khan, " US-GAN: on the importance of ultimate skip connection for facial expression synthesis ," *Multimedia Tools and Applications*, vol. 83, no. 3, pp. 7231–7247, June 2023, doi: 10.1007/s11042-023-15268-2.

[Zha+17] S. Zhang, B. Guo, A. Dong, J. He, Z. Xu, and S. X. Chen, "Cautionary tales on air-quality improvement in Beijing," *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 473, no. 2205, p. 20170457, 2017, doi: 10.1098/rspa.2017.0457.

[Mic21] Microsoft, "Neural Network Intelligence." Jan. 2021.

[Cro+16] M. Crosetto, O. Monserrat, M. Cuevas-González, N. Devanthéry, and B. Crippa, "Persistent Scatterer Interferometry: A Review," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 115, pp. 78–89, 2016, doi: 10.1016/j.isprsjprs.2015.10.011.

[Osm+16] B. Osmanoglu, F. Sunar, S. Wdowinski, and E. Cabral-Cano, "Time Series Analysis of InSAR Data: Methods and Trends," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 115, pp. 90–102, 2016, doi: 10.1016/j.isprsjprs.2015.10.003.

[Han01] R. F. Hanssen, *Radar Interferometry: Data Interpretation and Error Analysis*, vol. 2. Springer Science & Business Media, 2001.

[BH21] W. S. Brouwer and R. F. Hanssen, " An Analysis of Insar Displacement Vector Decomposition Fallacies and the Strap-down Solution ," in *2021 IEEE International Geoscience and Remote Sensing Symposium IGARSS*, 2021, pp. 2927–2930. doi: 10.1109/IGARSS47720.2021.9554216.

[CA21] C. Crippa and F. Agliardi, "Practical Estimation of Landslide Kinematics Using PSI Data," *Geosciences*, vol. 11, no. 214, 2021, doi: 10.3390/geosciences11050214.

[Ant+19] B. Antonielli, P. Mazzanti, A. Rocca, F. Bozzano, and L. Dei Cas, " A-DInSAR Performance for Updating Landslide Inventory in Mountain Areas: An Example from Lombardy Region (Italy) ," *Geosciences*, vol. 9, no. 364, 2019, doi: 10.3390/geosciences9090364.

[Guz+12] F. Guzzetti, A. Mondini, M. Cardinali, F. Fiorucci, M. Santangelo, and K.-t. Chang, " Landslide Inventory Maps: New Tools for an Old Problem. Earth Sci Rev 112(1-2):42-66 ," *Earth-Science Reviews*, vol. 112, pp. 42–66, Apr. 2012, doi: 10.1016/j.earscirev.2012.02.001.

[Maz+22] P. Mazzanti, S. Scancella, M. Virelli, S. Frittelli, V. Nocente, and F. Lombardo, " Assessing the Performance of Multi-Resolution Satellite SAR Images for Post-Earthquake Damage Detection and Mapping Aimed at Emergency Response Management ," *Remote Sensing*, vol. 14, no. 2210, 2022, doi: 10.3390/rs14092210.

[Sou+10] J. J. Sousa *et al.*, " PS-InSAR Processing Methodologies in the Detection of Field Surface Deformation—Study of the Granada Basin (Central Betic Cordilleras, Southern Spain) ," *Journal of Geodynamics*, vol. 49, no. 3, pp. 181–189, 2010, doi: 10.1016/j.jog.2009.12.002.

[MBM21] S. Moretto, F. Bozzano, and P. Mazzanti, " The Role of Satellite InSAR for Landslide Forecasting: Limitations and Openings ," *Remote Sensing*, vol. 13, no. 3735, 2021, doi: 10.3390/rs13183735.

[TS24] A. Tiwari and M. Shirzaei, " A Novel Machine Learning and Deep Learning Semi-Supervised Approach for Automatic Detection of InSAR-based Deformation Hotspots ," *International Journal of Applied Earth Observation and Geoinformation*, vol. 126, p. 103611, 2024, doi: 10.1016/j.jag.2023.103611.

[ASY15] S. Aghabozorgi, A. Seyed Shirkhorshidi, and T. Ying Wah, "Time-Series Clustering – a Decade Review," *Information Systems*, vol. 53, pp. 16–38, 2015, doi: 10.1016/j.is.2015.04.007.

[SS21] P. Schneider and U. Soergel, "Clustering Persistent Scatterer Points Based on a Hybrid Distance Metric," in *German Conference on Pattern Recognition*, 2021, pp. 621–632. doi: 10.1007/978-3-030-92659-5_40.

[ZZZ22] D. Zhou, X. Zuo, and Z. Zhao, " Constructing a Large-Scale Urban Land Subsidence Prediction Method Based on Neural Network Algorithm from the Perspective of Multiple Factors ," *Remote Sensing*, vol. 14, no. 8, p. 1803, 2022, doi: 10.3390/rs14081803.

[Jin+23] B. Jin *et al.*, " The Prediction of Transmission Towers' Foundation Ground Subsidence in the Salt Lake Area Based on Multi-Temporal Interferometric Synthetic Aperture Radar and Deep Learning ," *Remote Sensing*, vol. 15, no. 19, p. 4805, 2023, doi: 10.3390/rs15194805.

[Fes+23] D. Festa *et al.*, " Unsupervised Detection of InSAR Time Series Patterns Based on PCA and K-means Clustering ," *International Journal of Applied Earth Observation and Geoinformation*, vol. 118, p. 103276, 2023, doi: 10.1016/j.jag.2023.103276.

[Pen+24] M. Peng *et al.*, " Characterization and Prediction of InSAR-derived Ground Motion with ICA-assisted LSTM Model ," *Remote Sensing of Environment*, vol. 301, p. 113923, 2024, doi: 10.1016/j.rse.2023.113923.

[Zoc+25] M. Zocchi, C. Masciulli, G. Mastrantoni, F. Troiani, P. Mazzanti, and G. Scarascia Mugnozza, " Automatic Landslide Prioritization at Regional Scale through PS-InSAR Cluster Analysis and Socio-Economic Impacts ," *Remote Sensing Applications: Society and Environment*, vol. 37, p. 101414, 2025, doi: 10.1016/j.rsase.2024.101414.

[MS11] G. Milone and G. Scepi, " A Clustering Approach for Studying Ground Deformation Trends in Campania Region through PS-InSAR TM Time Series Analysis ," vol. 11, no. 4, pp. 610–620, 2011.

[Chi+24] S. D. Chicas, H. Li, N. Mizoue, T. Ota, Y. Du, and M. Somogyvári, " Landslide Susceptibility Mapping Core-Base Factors and Models' Performance Variability: A Systematic Review ," *Natural Hazards*, pp. 1–21, 2024.

[SN20] K. R. Shahapure and C. Nicholas, "Cluster Quality Analysis Using Silhouette Score," in *2020 IEEE 7th International Conference on Data Science and Advanced Analytics (DSAA)*, IEEE, 2020, pp. 747–748. doi: 10.1109/DSAA49011.2020.00096.

[PDG13] P. Pierantoni, G. Deiana, and S. Galdenzi, " Stratigraphic and Structural Features of the Sibillini Mountains (Umbria-Marche Apennines, Italy) ," *Italian Journal of Geosciences*, vol. 132, no. 3, pp. 497–520, 2013, doi: 10.3301/IJG.2013.08.

[Mar+15] M. Marini, S. Milli, R. Ravnås, and M. Moscatelli, " A Comparative Study of Confined vs. Semi-Confined Turbidite Lobes from the Lower Messinian Laga Basin (Central Apennines, Italy): Implications for Assessment of Reservoir Architecture ," *Marine and Petroleum Geology*, vol. 63, pp. 142–165, 2015, doi: 10.1016/j.marpetgeo.2015.02.015.

[Buc+21] F. Bucci *et al.*, "A New Digital Lithological Map of Italy at 1: 100.000 Scale," *PANGAEA*, 2021.

[Ama+20] M. Amanti *et al.*, " Geological and Geotechnical Models Definition for 3rd Level Seismic Microzonation Studies in Central Italy ," *Bulletin of Earthquake Engineering*, vol. 18, pp. 5441–5473, 2020, doi: 10.1007/s10518-020-00843-x.

[Mam+23] E. Mammoliti, D. Fronzi, S. Palpacelli, N. Biagiola, and A. Tazioli, " Assessment of Urban Landslide Groundwater Characteristics and Origin Using Artificial Tracers, Hydro-Chemical and Stable Isotope Approaches ," *Environmental Earth Sciences*, vol. 82, no. 9, p. 211, 2023, doi: 10.1007/s12665-023-10887-2.

[Gha+24] E. Ghaderpour, C. Masciulli, M. Zocchi, F. Bozzano, G. Scarascia Mugnozza, and P. Mazzanti, " Estimating Reactivation Times and Velocities of Slow-Moving Landslides via PS-InSAR and Their Relationship with Precipitation in Central Italy ," *Remote Sensing*, vol. 16, no. 16, p. 3055, 2024b, doi: 10.3390/rs16163055.

[Per+12] S. Peruccacci, M. T. Brunetti, S. Luciani, C. Vennari, and F. Guzzetti, " Lithological and Seasonal Control on Rainfall Thresholds for the Possible Initiation of Landslides in Central Italy ," *Geomorphology*, vol. 139, pp. 79–90, 2012, doi: 10.1016/j.geomorph.2011.10.005.

[Don+23] M. Donnini *et al.*, " Landslides Triggered by an Extraordinary Rainfall Event in Central Italy on September 15, 2022 ," *Landslides*, vol. 20, no. 10, pp. 2199–2211, 2023, doi: 10.1007/s10346-023-02109-4.

[Cro+20] M. Crosetto *et al.*, " The Evolution of Wide-Area DInSAR: From Regional and National Services to the European Ground Motion Service ," *Remote Sensing*, vol. 12, no. 2043, 2020, doi: 10.3390/rs12122043.

[Cos+21] M. Costantini *et al.*, "European Ground Motion Service (EGMS)," in *2021 IEEE International Geoscience and Remote Sensing Symposium IGARSS*, July 2021, pp. 3293–3296. doi: 10.1109/IGARSS47720.2021.9553562.

[Cro+21] M. Crosetto *et al.*, " Deformation Monitoring at European Scale: The Copernicus Ground Motion Service ," *The International Archives of the Photogrammetry,*

Remote Sensing and Spatial Information Sciences, pp. 141–146, 2021, doi: 10.5194/isprs-archives-XLIII-B3-2021-141-2021.

[Tar+07] S. Tarquini, I. Isola, M. Favalli, and A. Battistini, " TINITALY, a Digital Elevation Model of Italy with a 10 m-Cell Size (Version 1.0)[Data Set] ," *Istituto Nazionale di Geofisica e Vulcanologia (INGV)*, 2007.

[TN17] S. Tarquini and L. Nannipieri, " The 10 M-Resolution TINITALY DEM as a Trans-Disciplinary Basis for the Analysis of the Italian Territory: Current Trends and New Perspectives ," *Geomorphology*, vol. 281, pp. 108–115, 2017, doi: 10.1016/j.geomorph.2016.12.022.

[Tar+23] S. Tarquini, I. Isola, M. Favalli, A. Battistini, and G. T. Dotta, " A Digital Elevation Model of Italy with a 10 Meters Cell Size (Version 1.1) ," *Istituto Nazionale di Geofisica e Vulcanologia (INGV): Roma, Italy*, vol. 1, pp. 1–2, 2023.

[Fes+22] D. Festa *et al.*, " Automated Classification of A-DInSAR-based Ground Deformation by Using Random Forest ," *GIScience & Remote Sensing*, vol. 59, no. 1, pp. 1749–1766, 2022, doi: 10.1080/15481603.2022.2134561.

[Mil+19] P. Milillo *et al.*, " Heterogeneous Retreat and Ice Melt of Thwaites Glacier, West Antarctica ," *Science Advances*, vol. 5, Jan. 2019, doi: 10.1126/sciadv.aau3433.

[Sel+20] S. Selvakumaran *et al.*, "Combined InSAR and Terrestrial Structural Monitoring of Bridges," *IEEE Transactions on Geoscience and Remote Sensing*, pp. 1–13, Apr. 2020, doi: 10.1109/TGRS.2020.2979961.

[FAT21] S. S. Fikri, I. M. Anjasmara, and M. Taufik, " Application of Different Coherence Threshold on PS-InSAR Technique for Monitoring Deformation on the LUSI Affected Area during 2017 and 2019 ," *IOP Conference Series: Earth and Environmental Science*, vol. 731, 2021, doi: 10.1088/1755-1315/731/1/012036.

[Bar+17] A. Barra *et al.*, " A Methodology to Detect and Update Active Deformation Areas Based on Sentinel-1 SAR Images ," *Remote Sensing*, vol. 9, no. 1002, 2017, doi: 10.3390/rs9101002.

[Sol+19] L. Solari *et al.*, " A Sentinel-1 Based Hot-Spot Analysis: Landslide Mapping in North-Western Italy ," *International Journal of Remote Sensing*, vol. 40, Apr. 2019, doi: 10.1080/01431161.2019.1607612.

[Bis+20] C. Bischoff, A. Ferretti, F. Novali, A. Uttini, G. Chiara, and F. Meloni, " Nationwide Deformation Monitoring with SqueeSAR® Using Sentinel-1 Data ," *Proceedings of the International Association of Hydrological Sciences*, vol. 382, pp. 31–37, Apr. 2020, doi: 10.5194/piahs-382-31-2020.

[PG15] J. Paparrizos and L. Gravano, "K-Shape: Efficient and Accurate Clustering of Time Series," in *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*, 2015, pp. 1855–1870. doi: 10.1145/2723372.2737793.

[Hu+24] L. Hu, M. Jiang, J. Dong, X. Liu, and Z. He, "Interpretable Clustering: A Survey," *arXiv preprint arXiv:2409.00743*, 2024.

[TBN21] D. Tiano, A. Bonifati, and R. Ng, "FeatTS: Feature-Based Time Series Clustering," in *Proceedings of the 2021 International Conference on Management of Data*, in

Sigmod '21. New York, NY, USA: Association for Computing Machinery, 2021a, pp. 2784–2788. doi: 10.1145/3448016.3452757.

[Bon+22] A. Bonifati, F. D. Buono, F. Guerra, and D. Tiano, " Time2Feat: Learning Interpretable Representations for Multivariate Time Series Clustering ," *Proceedings of the VLDB Endowment*, vol. 16, no. 2, pp. 193–201, Oct. 2022, doi: 10.14778/3565816.3565822.

[Bon+23] A. Bonifati, F. Del Buono, F. Guerra, M. Lombardi, and D. Tiano, "Interpretable Clustering of Multivariate Time Series with Time2Feat," *Proceedings of the VLDB Endowment (PVLDB)*, vol. 16, no. 12, pp. 3994–3997, 2023, doi: 10.14778/3611540.3611604.

[CKF16] M. Christ, A. W. Kempa-Liehr, and M. Feindt, " Distributed and Parallel Time Series Feature Extraction for Industrial Big Data Applications ," *arXiv e-prints*, no. arXiv:1610.07717, 2016.

[Chr+18] M. Christ, N. Braun, J. Neuffer, and A. W. Kempa-Liehr, " Time Series Feature Extraction on Basis of Scalable Hypothesis Tests (Tsfresh–a Python Package) ," *Neurocomputing*, vol. 307, pp. 72–77, 2018, doi: 10.1016/j.neucom.2018.03.067.

[BP09] S. Bulò and M. Pelillo, "A Game-Theoretic Approach to Hypergraph Clustering," *Advances in neural information processing systems*, vol. 22, 2009.

[LLY10] H. Liu, L. Latecki, and S. Yan, "Robust Clustering as Ensembles of Affinity Relations," *Advances in neural information processing systems*, vol. 23, 2010.

[Leo20] M. Leordeanu, "Unsupervised Learning of Graph and Hypergraph Clustering," *Unsupervised Learning in Space and Time: A Modern Approach for Computer Vision using Graph-based Techniques and Deep Neural Networks*, pp. 107–124, 2020, doi: 10.1007/978-3-030-42128-1_3.

[Sch07] S. E. Schaeffer, "Graph Clustering," *Computer science review*, vol. 1, no. 1, pp. 27–64, 2007, doi: 10.1016/j.cosrev.2007.05.001.

[Liu+24] J. Liu *et al.*, " UniTST: Effectively Modeling Inter-Series and Intra-Series Dependencies for Multivariate Time Series Forecasting ," *arXiv e-prints*, no. arXiv:2406.04975, 2024b.

[RGP08] P. P. Rodrigues, J. Gama, and J. Pedroso, "Hierarchical Clustering of Time-Series Data Streams," *IEEE transactions on knowledge and data engineering*, vol. 20, no. 5, pp. 615–627, 2008, doi: 10.1109/TKDE.2007.190727.

[Jol18] K. Jolly, *Machine Learning with Scikit-Learn Quick Start Guide: Classification, Regression, and Clustering Techniques in Python*. Packt Publishing Ltd, 2018.

[HK11] J. Hauke and T. Kossowski, " Comparison of Values of Pearson's and Spearman's Correlation Coefficients on the Same Sets of Data ," *Quaestiones geographicae*, vol. 30, no. 2, pp. 87–93, 2011, doi: 10.2478/v10117-011-0021-1.

[SBS18] P. Schober, C. Boer, and L. A. Schwarte, "Correlation Coefficients: Appropriate Use and Interpretation," *Anesthesia & analgesia*, vol. 126, no. 5, pp. 1763–1768, 2018, doi: 10.1213/ANE.0000000000002864.

[Jas+22] A. Jastrzebska, G. Nápoles, Y. Salgueiro, and K. Vanhoof, "Evaluating Time Series Similarity Using Concept-Based Models," *Knowledge-Based Systems*, vol. 238, p. 107811, 2022, doi: 10.1016/j.knosys.2021.107811.

[Lu+07] Y. Lu, I. Cohen, X. S. Zhou, and Q. Tian, "Feature Selection Using Principal Feature Analysis," in *Proceedings of the 15th ACM International Conference on Multimedia*, in Mm '07. New York, NY, USA: Association for Computing Machinery, 2007, pp. 301–304. doi: 10.1145/1291233.1291297.

[SGM10] F. Song, Z. Guo, and D. Mei, "Feature Selection Using Principal Component Analysis," in *2010 International Conference on System Science, Engineering Design and Manufacturing Informatization*, 2010, pp. 27–30. doi: 10.1109/ICSEM.2010.14.

[TBN21] D. Tiano, A. Bonifati, and R. Ng, "Feature-Driven Time Series Clustering," in *24th International Conference on Extending Database Technology, EDBT 2021*, 2021b, pp. 349–354.

[GMG24] H. Guo, A. M. Mart\inez-Graña, and J. A. González-Delgado, " Monitoring the Subsidence in Wan'an Town of Deyang Based on PS-InSAR Technology (Sichuan, China) ," *Sustainability*, vol. 16, no. 22, p. 10010, 2024, doi: 10.3390/su162210010.

[HB08] L. M. Highland and P. Bobrowsky, *The Landslide Handbook-a Guide to Understanding Landslides*, no. 1325. US Geological Survey, 2008.

[Zhu+22] Y. Zhu, X. Yao, C. Yao, Z. Zhou, Z. Gu, and L. Yao, " Integration of Vertical and Horizontal Deformation Derived by SAR Observation for Identifying Landslide Motion Patterns in a Basaltic Weathered Crust Region of Guizhou, China ," *Remote Sensing*, vol. 14, no. 16, p. 4014, 2022, doi: 10.3390/rs14164014.

[AR23] K. Andrii and S. Roman, " Methods of Cluster Analysis for Detection of Uniform Displacement Zones of Landslides and Anti-Landslide Structures ," *Journal of Basic & Applied Sciences*, vol. 19, pp. 151–162, 2023, doi: 10.29169/1927-5129.2023.19.13.

[Fra+18] P. Frattini, G. B. Crosta, M. Rossini, and J. Allievi, " Activity and Kinematic Behaviour of Deep-Seated Landslides from PS-InSAR Displacement Rate Measurements ," *Landslides*, vol. 15, pp. 1053–1070, 2018, doi: 10.1007/s10346-017-0940-6.

[Cri+21] C. Crippa, E. Valbuzzi, P. Frattini, G. B. Crosta, M. C. Spreafico, and F. Agliardi, " Semi-Automated Regional Classification of the Style of Activity of Slow Rock-Slope Deformations Using PS InSAR and SqueeSAR Velocity Data ," *Landslides*, vol. 18, pp. 2445–2463, 2021, doi: 10.1007/s10346-021-01654-0.

[Bia+18] S. Bianchini *et al.*, " From Picture to Movie: Twenty Years of Ground Deformation Recording over Tuscany Region (Italy) with Satellite InSAR ," *Frontiers in Earth Science*, vol. 6, p. 177, 2018, doi: 10.3389/feart.2018.00177.

[Wan+24] X. Wang, X. Wang, H. Guo, and A. Zhang, " Unsupervised Deep Clustering Method for Coseismic Landslide Recognition Based on High-Resolution Images and Implicit Knowledge ," *IEEE Transactions on Geoscience and Remote Sensing*, 2024c.

[Guo+06] D. Guo, J. Chen, A. M. MacEachren, and K. Liao, " A Visualization System for Space-Time and Multivariate Patterns (Vis-Stamp) ," *IEEE transactions on visualization and computer graphics*, vol. 12, no. 6, pp. 1461–1474, 2006, doi: 10.1109/TVCG.2006.84.

[SKS20] P. Schneider, R. Khamis, and U. Soergel, " Extracting and Evaluating Clusters in Dinsar Deformation Data on Single Buildings ," *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 3, pp. 157–163, 2020.

[Rig+23] S. Rigamonti, G. Dattola, P. Frattini, and G. B. Crosta, " A Multivariate Time Series Analysis of Ground Deformation Using Persistent Scatterer Interferometry ," *Remote Sensing*, vol. 15, no. 12, p. 3082, 2023, doi: 10.3390/rs15123082.

[Fer+20] L. N. Ferreira *et al.*, "Spatiotemporal Data Analysis with Chronological Networks," *Nature Communications*, vol. 11, p. 4036, Aug. 2020, doi: 10.1038/s41467-020-17634-2.

[Bel+97] A. Belouchrani, K. Abed-Meraim, J.-F. Cardoso, and E. Moulines, "A blind source separation technique using second-order statistics," *IEEE Transactions on Signal Processing*, vol. 45, no. 2, pp. 434–444, 1997, doi: 10.1109/78.554307.

[Gad+18] M. Gaddes, A. Hooper, M. Bagnardi, H. Inman, and F. Albino, " Blind Signal Separation Methods for InSAR: The Potential to Automatically Detect and Monitor Signals of Volcanic Deformation ," *Journal of Geophysical Research: Solid Earth*, vol. 123, no. 11, pp. 10–226, 2018, doi: 10.1029/2018JB016210.

[Agu+21] P. Aguiar, A. Cunha, M. Bakon, A. M. Ruiz-Armenteros, and J. J. Sousa, " Multivariate Outlier Detection in Postprocessing of Multi-Temporal PS-InSAR Results Using Deep Learning ," *Procedia Computer Science*, vol. 181, pp. 1146–1153, 2021, doi: 10.1016/j.procs.2021.01.326.

[CBC13] F. Cigna, S. Bianchini, and N. Casagli, " How to Assess Landslide Activity and Intensity with Persistent Scatterer Interferometry (PSI): The PSI-based Matrix Approach ," *Landslides*, vol. 10, pp. 267–283, 2013, doi: 10.1007/s10346-012-0335-7.

[Din+25] P. Dinklage, J. Fischer, L. Nalbach, and J. Zumbrink, "RLZ-r and LZ-End-r: Enhancing Move-r." [Online]. Available: https://arxiv.org/abs/2507.17300