ShareDAB-DETR: Memory-Efficient Object Detection through CentralBank Parameter Sharing

Khaled Chikh[a], Roberto Cavicchioli[b] and Alessandro Capotondi[c]
Department of Physics, Informatics and Mathematics, University of Modena and Reggio Emilia, Italy

Keywords: Parameter Sharing, Object Detection, Vision Transformers, DETR, Edge Deployment, Model Compression.

Abstract: Transformer-based object detectors achieve state-of-the-art accuracy but require substantial memory, making deployment on edge devices impractical. We introduce CentralBank, a novel parameter sharing mechanism that consolidates all transformer projection weights into a single memory-efficient table accessed through deterministic offset mapping. Applied to DAB-DETR, our approach achieves up to 84% reduction in transformer parameters (19.67M to 3.21M) and 67% reduction in training memory while retaining 99% of baseline accuracy (41.8 vs 42.2 AP on COCO). Unlike traditional compression methods, CentralBank is integrated at the architectural level, requiring no post-training optimization. Our best configuration achieves 41.8 AP with only 8.65M transformer parameters and 2.5 GB training memory, enabling practical deployment on edge devices. Extensive evaluation across five configurations reveals that larger hidden dimensions (512 vs 256) provide superior parameter efficiency, achieving an 11.2:1 parameter-to-FLOP reduction ratio. ShareDAB-DETR demonstrates that architectural parameter sharing can effectively bridge the gap between transformer performance and edge deployment constraints.

1 INTRODUCTION

Vision transformers have revolutionized object detection, with DETR-based architectures (Carion et al., 2020; Liu et al., 2022; Zhu et al., 2021b) achieving state-of-the-art accuracy through end-to-end detection learning. However, these object detection models incur significant computational costs. DAB-DETR with ResNet-50 backbone requires 44M parameters and 6.5GB GPU memory at batch size 2, rendering it unsuitable for edge devices like NVIDIA Jetson Nano (4GB RAM) or embedded GPUs in autonomous vehicles where real-time object detection is essential.

The core inefficiency stems from redundancy in transformer layers: each of six encoder and six decoder layers maintains separate projection matrices for queries, keys, values, and feed-forward networks. This architectural choice prioritizes model capacity over efficiency, resulting in 19.67M transformer parameters alone, nearly half the total model size. Although this redundancy provides flexibility during training, much of it proves unnecessary for object detection inference, creating opportunities for compression without sacrificing accuracy (Xu et al., 2024a; Frankle et al., 2019).

Existing compression techniques address this inefficiency through post-training optimization such as pruning (Han et al., 2016) where redundant weights are removed, quantization (Moon et al., 2024) to reduce precision, and distillation (Habib et al., 2024) transfers knowledge to smaller detection models. However, these methods introduce complexity, require careful hyperparameter tuning, and often degrade detection accuracy. Moreover, they operate after training is complete, missing opportunities for efficiency-aware learning from initialization.

Parameter sharing offers a different strategy by merging redundant parameters during training rather than removing them afterward. In natural language processing (NLP), models such as ALBERT (Lan et al., 2020) and ShareBERT (Hu et al., 2024) successfully demonstrate effective layer-wise weight sharing in transformers. However, extending these ideas to vision object detection is nontrivial: ShareBERT's embedding-based sharing introduces computational overhead (126 vs 28 GFLOPs for BERT), and both methods enforce dimensional restrictions that do not align with the demands of high-resolution object

[a] https://orcid.org/0009-0009-9308-0380
[b] https://orcid.org/0000-0003-0166-0898
[c] https://orcid.org/0000-0001-8705-0761

Chikh, K., Cavicchioli, R. and Capotondi, A.
ShareDAB-DETR: Memory-Efficient Object Detection through CentralBank Parameter Sharing.
DOI: 10.5220/0014452400004084

In *Proceedings of the 21st International Conference on Computer Vision Theory and Applications (VISAPP 2026) - Volume 1*, pages 222-233
ISBN: 978-989-758-804-4; ISSN: 2184-4321

detection in vision.

To address these issues we introduce *CentralBank*, a novel parameter sharing mechanism designed specifically for vision transformers in object detection. CentralBank consolidates all projection weights into a single contiguous memory block and dynamically constructs layer-specific weights through deterministic offset-based slicing. Unlike NLP approaches, CentralBank eliminates dimensional constraints, avoids computational overhead through zero-copy tensor views, and applies uniformly to all transformer components.

Our contributions are: (1) CentralBank, a unified parameter sharing mechanism that eliminates redundancy in detection transformers through architectural design rather than post-training compression; (2) comprehensive analysis revealing that larger hidden dimensions combined with aggressive sharing yield superior parameter efficiency compared to smaller dimensions; (3) extensive evaluation across five ShareDAB-DETR variants demonstrating favorable accuracy-efficiency trade-offs for object detection, with practical guidance for deployment scenarios; (4) empirical demonstration that 67% memory reduction enables training and deployment of DETR detectors on resource-constrained edge devices while maintaining near-baseline detection accuracy on COCO.

The remainder of this paper is structured as follows: Section 2 reviews DETR architectures and object detection compression methods, Section 3 presents the CentralBank mechanism, Section 4 describes ShareDAB-DETR, Sections 5 detail experimental evaluation on COCO, Section 6 analyzes key findings, and Section 7 concludes and Section 8 outlines future work.

2 RELATED WORK

This section positions CentralBank within the broader landscape of vision transformer efficiency. We organize our discussion across four complementary dimensions. First, we review recent DETR-based object detection architectures that define the accuracy frontier and establish the memory inefficiency we address. Second, we survey traditional model compression techniques such as pruning, quantization, distillation, and low-rank decomposition and their fundamental deployment constraints. Third, we examine parameter sharing approaches developed primarily for NLP transformers, identifying critical limitations when applied to vision. Finally, we articulate the architectural mismatches that motivate CentralBank's design. This structure reveals why existing compression paradigms fall short for edge-constrained vision transformers and why architectural parameter sharing offers a fundamentally different path forward.

2.1 DETR-Based Object Detection

Beyond the original DETR (Carion et al., 2020), which achieved 42.0 AP but required 500 training epochs, the field has progressively addressed convergence and accuracy through various architectural approaches. Table 1 surveys this evolution.

Several works focused on attention mechanisms to accelerate training. Deformable DETR (Zhu et al., 2021b) concentrates attention on sparse sampling locations, achieving 43.8 AP with 10× faster convergence (50 epochs). Conditional DETR (Meng et al., 2021) takes a different approach with conditional spatial queries, reaching 43.0 AP in the same 50 epochs.

Query design has also proven important. Anchor DETR (Wang et al., 2022) initializes queries as anchor points borrowed from CNN detectors, while DAB-DETR (Liu et al., 2022) extends this to 4D dynamic anchor boxes refined across layers. DAB-DETR achieves 42.2 AP as a strong baseline, yet its 6527 MB training memory footprint creates practical deployment constraints.

More recent efforts address training stability. DN-DETR (Li et al., 2022) introduces denoising training alongside standard detection, reaching 48.6 AP a substantial +6.4 improvement. DINO (Zhang et al., 2023) pushes further with contrastive denoising and mixed query selection, achieving 49.4 AP in just 12 epochs. Other work like H-Deformable-DETR (Jia et al., 2023) and Group DETR (Chen et al., 2023) explore hybrid and group-wise assignment strategies, achieving 48.7 and 46.8 AP respectively.

The field has achieved impressive accuracy gains 49.4 vs 42.0 AP but memory remains largely unaddressed. Model parameters stay around 40-44M across variants, and training memory is rarely reported. Our approach takes a different path: rather than chasing peak accuracy, we tackle the memory bottleneck through architectural parameter sharing, achieving 67% memory reduction (6527 to 2503 MB) with just 0.4 AP degradation and comparable computational cost (89 GFLOPs).

2.2 Model Compression for Transformers

Model compression for transformers addresses the computational demands of these architectures through distinct technical approaches. We organize this dis-

Table 1: DETR-based architectures on COCO with ResNet-50 backbone. Our best ShareDAB-DETR variant achieves substantial memory reduction while maintaining competitive accuracy.

Method	Epochs	AP	Params (M)	GFLOPs	Memory (MB)
DETR (Carion et al., 2020)	500	42.0	41.0	86	–
Deformable DETR (Zhu et al., 2021b)	50	43.8	40.0	173	–
Conditional DETR (Meng et al., 2021)	50	43.0	44.0	90	–
Anchor DETR (Wang et al., 2022)	50	42.1	–	–	–
DAB-DETR (Liu et al., 2022)	50	42.2	44.0	94	6527
DN-DETR (Li et al., 2022)	50	48.6	44.0	–	–
DINO (Zhang et al., 2023)	12	49.4	44.0	–	–
H-Deformable-DETR (Jia et al., 2023)	50	48.7	–	–	–
Group DETR (Chen et al., 2023)	50	46.8	–	–	–
ShareDAB-DETR (Ours)	50	41.8	35.8	89	2503

cussion by compression methodology, analyzing how each technique applies to vision transformers and identifying their limitations for our target application.

Pruning reduces model complexity by eliminating redundant parameters. Structured pruning methods remove entire network components while maintaining regular computation patterns. PatchSlimming (Tang et al., 2022) progressively discards uninformative image patches using top-down estimation, achieving 40% FLOPs reduction on vision transformers. X-Pruner (Yu and Xiang, 2023) introduces explainability-aware pruning through learnable masks that measure each unit's contribution to target classes, enabling 51-66% FLOPs reduction with minimal accuracy loss. SAViT (Zheng et al., 2022) integrates structural interactions for collaborative pruning across multiple dimensions, while VTP (Zhu et al., 2021a) applies dimension pruning primarily to feed-forward blocks. However, pruned models require specialized sparse computation kernels for deployment, limiting their practical applicability.

Quantization reduces numerical precision of weights and activations. PTQ4ViT (Yuan et al., 2024) addresses the challenge of long-tail distributions in post-Softmax and post-GELU activations through twin uniform quantization coupled with Hessian-guided metric optimization. FQ-ViT (Lin et al., 2023) quantizes all linear and self-attention layers along with LayerNorm and SoftMax operations using power-of-two factors and Log2 quantizers. OmniQuant (Shao et al., 2024) achieves low-bit quantization (W4A4, W3A16, W2A16) through learnable weight clipping and equivalent transformation, optimizing via block-wise error minimization (Xu et al., 2024b; Yuan et al., 2024; Wu et al., 2024; Wu et al., 2025). Recent advances include RepQ-ViT (Li et al., 2023) with channel-wise quantization and reparameterization, and ERQ (Zhong et al., 2025) which sequentially reduces quantization errors. Despite effectiveness, quantized models demand specialized hardware support (INT4/INT8 accelerators), constraining deployment flexibility.

Knowledge distillation transfers knowledge from large teacher models to compact student models. DeiT (Touvron et al., 2021) pioneered transformer-specific distillation by introducing a distillation token alongside the class token, enabling efficient training without large-scale datasets. DistilBERT (Sanh et al., 2020) demonstrates that a 40% smaller model retains 97% of BERT's performance using triple loss combining masked language modeling, distillation loss, and cosine embedding loss. Advanced strategies include layer-wise distillation matching representations between layers, attention-based distillation transferring attention patterns, and multi-crop distillation with oscillation-aware regularization (Xu et al., 2024b; Liu et al., 2024; Lu et al., 2022; Brown et al., 2023; Liu, 2022). Distillation methods require maintaining teacher models during training, increasing computational overhead and storage requirements.

Low-rank decomposition factorizes weight matrices using techniques such as Singular Value Decomposition. Progressive decomposition strategies achieve superior accuracy-compression tradeoffs compared to single-shot approaches (Dong et al., 2024; Kang et al., 2025; Chang et al., 2024). LRA-QViT (Kang et al., 2025) integrates low-rank approximation with quantization, introducing reparameterizable branch-based decomposition and weight reconstruction, achieving 31.8% parameter reduction with 0.03% accuracy drop on Swin-B. Low-rank methods can struggle with activation outliers and require careful rank selection.

Unified compression frameworks combine multiple techniques into joint optimization. UVC (Yu et al., 2022) integrates pruning, layer skipping, and knowledge distillation into constrained optimization solved via primal-dual algorithms, achieving 50% FLOPs reduction with 0.3% accuracy loss on DeiT-

Tiny. UPop (Shi et al., 2023) progressively searches multimodal subnets with adaptive compression ratio assignment. Token Merging (ToMe) (Bolya et al., 2022) gradually merges similar tokens achieving 2-3× speedups, while MobileViT (Mehta and Rastegari, 2021) combines CNN local processing with transformer global modeling for mobile deployment (Liu et al., 2025; Xu et al., 2024b).

2.3 Parameter Sharing in Transformers

Parameter sharing offers an alternative compression paradigm by reusing weights across layers, fundamentally reducing memory footprint without requiring post-training optimization or specialized hardware. However, existing parameter sharing approaches were designed primarily for NLP transformers and face critical limitations when applied to vision transformers.

Cross-layer parameter sharing reuses the same weights across multiple transformer layers. ALBERT (Lan et al., 2020) applies uniform parameter sharing across all layers through factorization, reducing BERT's parameter count by 18× and enabling 1.7× faster training. This approach shares both attention and feedforward parameters uniformly across depth. However, analysis reveals that uniform sharing prevents parameter growth with network depth and affects gradient value ranges during training (Lan et al., 2020; Xie et al., 2021). Layer Tying methods (Hay and Wolf, 2024; Bae et al., 2024) employ reinforcement learning to adaptively determine which layers share weights, reducing trainable parameters and memory by up to one order of magnitude. Recursive Transformers (Bae et al., 2024) convert standard models into compact versions reusing single blocks multiple times, enhanced through depth-wise LoRA modules. While effective for NLP, these methods do not address the architectural differences between text and vision transformers.

Embedding-based parameter sharing constructs layer parameters from embedding matrices. ShareBERT (Hu et al., 2024) leverages embedding matrices to construct hidden layer parameters through column sharing, achieving 21.9× compression (5M versus BERT's 110M parameters) while preserving 95.5% of accuracy. The method employs Embedding Parameter Sharing (EPS) which directly shares embedding dimensions with layers, and Virtual Embedding Parameter Sharing (VEPS) for cases when required dimensions exceed vocabulary size. This approach faces two critical limitations: first, it constrains weight dimensions to vocabulary size (typically 30K tokens for BERT), unsuitable for vision transformer projection matrices where hidden dimensions (768, 1024) vastly exceed vocabulary constraints; second, reconstructing weights from shared embeddings at each forward pass incurs 4.5× computational overhead (126 versus 28 GFLOPs), negating parameter reduction benefits.

2.4 Limitations for Vision Transformers

The reviewed compression methods face fundamental limitations when applied to vision transformers. All pruning, quantization, distillation, and low-rank methods operate after training, thereby missing opportunities for efficiency-aware learning during initialization and requiring additional optimization procedures. In addition, these approaches introduce specific deployment constraints: pruned models rely on sparse computation kernels, quantized models demand specialized hardware such as INT4 or INT8 accelerators, distillation requires maintaining teacher models, and low-rank methods often fail to handle outlier sensitivity. Moreover, embedding-based parameter sharing techniques originally developed for natural language processing exhibit architectural mismatches when applied to vision transformers. Their vocabulary-constrained dimensions are unsuitable for arbitrary projection matrices, they incur additional computational cost due to weight reconstruction, and they cannot efficiently manage the quadratic scaling of spatial dimensions (for example, $14 \times 14 = 196$ patches in images compared with 512 tokens in BERT). Finally, vision transformers operate on high-dimensional continuous features derived from image patches rather than on discrete embeddings, which calls for alternative parameter-sharing strategies that can preserve spatial feature relationships.

CentralBank addresses these limitations by embedding efficiency into the architecture from initialization, requiring no post-training optimization and imposing no deployment constraints beyond standard transformer operations. Unlike embedding-based sharing, our approach handles arbitrary projection dimensions without computational overhead, and unlike cross-layer sharing, we introduce controlled parameter sharing that maintains model expressiveness while achieving compression.

3 CENTRALBANK METHOD

We propose ShareDAB-DETR, illustrated in Figure 1, which consolidates all transformer projection weights into a unified CentralBank table via deterministic offset mapping. This design abstracts conventional

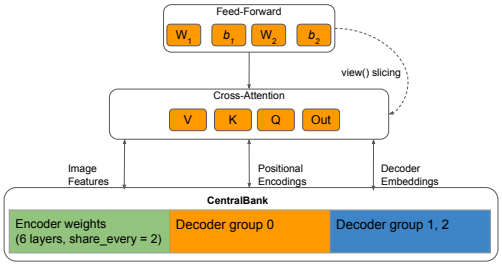

Figure 1: ShareDAB-DETR architecture overview. For clarity, we only show the cross-attention part in the Transformer decoder. All projection weights are accessed via CentralBank , enabling memory-efficient sharing.

layer-specific parameters and provides efficient views for transformer computation.

3.1 Motivation and Design Principles

Transformer layers exhibit substantial parameter redundancy. In DAB-DETR's 6-layer encoder-decoder, each layer maintains separate Q/K/V projections and feed-forward networks, totaling 19.67M parameters. However, learned representations across layers show high similarity, suggesting many parameters encode redundant transformations. Traditional approaches address this through post-training compression, but we propose architectural sharing that eliminates redundancy from initialization.

CentralBank operates on three design principles: *(1) Unified storage*: all projection weights reside in a single contiguous parameter table, eliminating per-layer instantiation overhead; *(2) Dynamic construction*: weights are constructed on-demand through tensor slicing rather than stored explicitly, enabling zero-copy memory access; *(3) Deterministic mapping*: offset-based indexing ensures collision-free parameter access and reproducible training.

3.2 CentralBank Architecture

CentralBank is implemented as a learnable parameter table $\mathbf{B} \in \mathbb{R}^{N \times d}$, where d is the hidden dimension and N is computed to accommodate all shared projection parameters. Each projection matrix $\mathbf{W}_i \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$ is constructed by slicing and reshaping a segment from $\mathbf{B}$:

$$\mathbf{W}_i = \text{reshape}(\mathbf{B}[o_i : o_i + d_{\text{out}} \cdot d_{\text{in}}], [d_{\text{out}}, d_{\text{in}}]), \quad (1)$$

where o_i is the pre-computed offset for projection i. This design provides several advantages: (i) all parameters reside in unified memory, improving cache locality; (ii) tensor views avoid data copying, maintaining constant memory regardless of sharing configuration; (iii) gradients accumulate directly into $\mathbf{B}$, simplifying backpropagation.

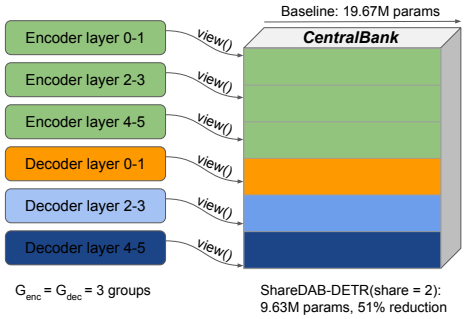

Figure 2: CentralBank group sharing. Each group of transformer layers maps to contiguous segments in the shared parameter table.

To balance compression and model capacity, we partition layers into groups of size g that share projections. For encoder layers with group size g_{enc} and decoder layers with g_{dec}, the number of unique projection sets is:

$$G = \left\lceil \frac{L_{\text{enc}}}{g_{\text{enc}}} \right\rceil + \left\lceil \frac{L_{\text{dec}}}{g_{\text{dec}}} \right\rceil \quad (2)$$

For example, $g_{\text{enc}} = g_{\text{dec}} = 6$ creates two groups (one encoder, one decoder) sharing parameters across all layers, maximizing compression. Conversely, $g_{\text{enc}} = g_{\text{dec}} = 2$ creates six groups, preserving more capacity at the cost of parameters.

A key aspect of CentralBank is its flexible parameter grouping, mapped schematically in Figure 2. Each transformer block accesses specific slices from the shared table according to the offset plan, allowing joint optimization of storage and expressiveness.

3.3 Dynamic Weight Construction

We implement CentralBank through ShareLinearDynamic, a lightweight wrapper replacing standard linear projections. During forward pass, ShareLinearDynamic performs three operations: (1) retrieve precomputed offset o_i from the deterministic map; (2) create a tensor view into $\mathbf{B}$ without copying data; (3) reshape the 1D view into required 2D weight matrix. Given input $\mathbf{x} \in \mathbb{R}^{b \times d_{\text{in}}}$, the output is:

$$\mathbf{y} = \mathbf{x} \cdot \mathbf{W}_i^{\top} + \mathbf{b}_i, \quad (3)$$

where $\mathbf{b}_i$ is an optional bias term also sliced from $\mathbf{B}$.

The key innovation is *view-based slicing*: PyTorch's tensor view creates a reference to the original memory rather than copying data. This zero-copy operation ensures that regardless of how weights are shared, memory consumption remains constant.

Moreover, autograd treats views as differentiable with respect to the original tensor, enabling gradients to accumulate directly into **B** during backpropagation. This design yields approximately 2× lower memory usage compared to ShareBERT's reconstruction approach, directly contributing to our observed 67% memory reduction.

3.4 Deterministic Offset Mapping

To ensure collision-free parameter access, CentralBank assigns each projection a unique contiguous segment through deterministic offset computation. For a group j containing projections $\{i_1, i_2, \ldots, i_k\}$, the offset for projection i_m is:

$$o_{i_m,j} = j \cdot S_{\text{group}} + \sum_{n=1}^{m-1} (d_{\text{out}}^{(i_n)} \cdot d_{\text{in}}^{(i_n)}), \tag{4}$$

where $S_{\text{group}} = \sum_{i \in \text{group}} (d_{\text{out}}^{(i)} \cdot d_{\text{in}}^{(i)})$ is the total parameters per group. This formulation guarantees: (i) disjoint memory ranges for each group; (ii) consistent projection ordering within groups; (iii) reproducible training across runs and hardware.

The offset map is computed once during initialization and reused throughout training and inference. This eliminates dynamic memory allocation overhead and enables compiler optimizations for cache-efficient access patterns—critical for edge deployment where memory bandwidth is constrained.

4 SHAREDAB-DETR IMPLEMENTATION

We apply CentralBank to DAB-DETR by systematically replacing all nn.Linear modules in encoder, decoder, and cross-attention blocks with ShareLinearDynamic wrappers. This substitution requires no changes to attention mechanisms, feed-forward networks, or matching heads, demonstrating CentralBank's model-agnostic nature.

We evaluate five configurations exploring the compression-accuracy trade-off space:
2-2-256: Shares every 2 encoder and decoder layers ($g_{\text{enc}} = g_{\text{dec}} = 2$), creating three groups per component. This conservative sharing preserves substantial capacity while achieving moderate compression.
3-2-256: Asymmetric sharing with $g_{\text{enc}} = 3, g_{\text{dec}} = 2$, exploring whether encoder and decoder benefit from different sharing granularities.
3-3-256: Shares every 3 layers ($g_{\text{enc}} = g_{\text{dec}} = 3$), balancing compression and capacity with two groups per component.
6-6-256: Aggressive sharing across all 6 layers ($g_{\text{enc}} = g_{\text{dec}} = 6$), creating single groups for encoder and decoder. Maximizes compression but may limit model capacity.
6-6-512: Combines aggressive sharing ($g_{\text{enc}} = g_{\text{dec}} = 6$) with increased hidden dimension (512 vs 256). Tests whether larger dimensions can offset capacity loss from extreme sharing.

These configurations span compression ratios from 22% (2-2-256) to 37% (6-6-256) for total parameters, and 51% to 84% for transformer parameters specifically. The diversity enables analysis of how sharing granularity and hidden dimension interact to determine the parameter-accuracy frontier.

5 EXPERIMENTAL EVALUATION

This section describes our comprehensive evaluation framework designed to assess CentralBank's parameter-accuracy trade-offs across compression levels, computational efficiency, and practical deployment scenarios. We detail our experimental setup, benchmark choices, and systematic evaluation protocol that reveals when and how parameter sharing delivers both memory efficiency and competitive accuracy.

5.1 Experimental Setup

All ShareDAB-DETR variants are trained on COCO 2017 for 50 epochs using 4 NVIDIA A100 GPUs (batch size 16, 4 per GPU). The backbone is ResNet-50 with standard COCO augmentation, following DAB-DETR's training protocol: AdamW optimizer (learning rate 1e-4, backbone 1e-5), weight decay 1e-4, and learning rate reduction at epoch 40. Input images are resized to 800×1200 pixels. We measure standard COCO metrics (AP, AP50, AP75, APS, APM, APL) and efficiency metrics (parameters, memory, latency, throughput) on a single NVIDIA RTX 3070 GPU. Latency measurements are averaged over 100 inference passes with batch size 1, and throughput is measured across batch sizes 1, 2, and 4 to characterize inference scalability.

5.2 Accuracy and Efficiency Evaluation

Table 2 presents comprehensive evaluation across all five ShareDAB-DETR configurations on COCO 2017 val set, reporting parameter counts, accuracy metrics, and memory consumption. ShareDAB-6-6-512 achieves 41.8 AP with 8.65M transformer parameters (56% reduction from baseline 19.67M) and 2503

MB training memory, maintaining 99% accuracy retention compared to baseline 42.2 AP. In contrast, ShareDAB-6-6-256 demonstrates the most aggressive compression with 3.21M transformer parameters (83.7% reduction) and 2164 MB memory, sacrificing more accuracy at 92.9% retention. Intermediate configurations (ShareDAB-2-2-256, ShareDAB-3-2-256) achieve 40.1 AP with 51-58% transformer parameter reduction and 95.0% accuracy retention.

Training memory exhibits consistency across configurations, ranging from 2164 MB (ShareDAB-6-6-256) to 2503 MB (ShareDAB-6-6-512). This range represents substantial reduction from baseline 6527 MB, enabling deployment on 4GB GPU devices across all variants. Parameter reduction ranges from 18.6% (ShareDAB-6-6-512) to 36.6% (ShareDAB-6-6-256) for total model parameters, with transformer-specific reductions reaching 51-84%.

Table 3 provides detailed breakdown of COCO detection metrics alongside inference efficiency measurements. Latency and throughput measurements reveal distinct inference characteristics across configurations. ShareDAB-6-6-512 exhibits higher single-image latency (89.75 ms at batch 2) compared to baseline (68.18 ms), representing a 31.6% increase. This reflects the larger hidden dimension (512 vs. 256), which increases per-token computational cost. In contrast, ShareDAB-6-6-256 achieves 71.67 ms latency at batch 2, only 5.2% higher than baseline, demonstrating that aggressive parameter sharing with smaller dimensions enables faster inference.

Throughput analysis demonstrates consistent scaling behavior across batch sizes. At batch 2, ShareDAB-2-2-256, ShareDAB-3-2-256, and ShareDAB-6-6-256 maintain throughput within 8-10% of baseline (13.31-13.95 imgs/s vs. 14.67 baseline). ShareDAB-6-6-512 exhibits reduced throughput (11.14 imgs/s, 24% below baseline), consistent with the larger hidden dimension. Throughput scaling from batch 1 to batch 4 remains stable across configurations, indicating that parameter sharing does not introduce batch-processing inefficiencies.

5.3 Trade-off Visualization

Figure 3 visualizes the parameter-accuracy trade-off landscape. The left panel traces progressive transformer parameter reduction from baseline 19.67M to 3.21M (ShareDAB-6-6-256), an 84% reduction. The right panel plots accuracy retention against transformer parameter reduction percentage, revealing ShareDAB-6-6-512 achieving 99% retention with 56% reduction and ShareDAB-6-6-256 reaching 84% reduction at 93% retention.

Three-dimensional efficiency analysis, presented in Figure 4, examines the relationship between transformer parameters, computational cost (GFLOPs), and accuracy retention. ShareDAB-6-6-512 occupies a region with 8.65M parameters, 89.3 GFLOPs (3.0% reduction from 92.1 baseline), and 99% accuracy retention. In contrast, ShareDAB-6-6-256 achieves greater parameter reduction (3.21M parameters, 11.6% FLOP reduction) with 92.9% accuracy retention. The variance between parameter and FLOP reduction reveals distinct scaling characteristics: parameter sharing in the transformer achieves greater efficiency gains than computational savings due to backbone dominance.

Parameter efficiency analysis reveals FLOP reduction ranging from 3.0% (ShareDAB-6-6-512) to 11.6% (ShareDAB-256 variants). This modest FLOP reduction reflects ResNet-50 backbone dominance: the backbone accounts for approximately 67 of 92.1 GFLOPs, leaving only 25.1 GFLOPs for transformer operations. Consequently, parameter sharing optimization in the transformer yields limited overall computational reduction, as shown in Figure 5.

5.4 Deployment Scenario Characterization

Figure 6 characterizes the parameter-accuracy trade-off space for deployment selection. The 95% AP threshold (green line) delineates configurations suitable for production deployment (above threshold) from aggressive compression regimes (below threshold). Configurations above this threshold include ShareDAB-2-2-256, ShareDAB-3-2-256, and ShareDAB-6-6-512, all achieving 40.1 AP or greater. Configurations below the threshold (ShareDAB-3-3-256 at 94.3%, ShareDAB-6-6-256 at 92.9%) represent extreme compression scenarios viable when accuracy trade-offs are acceptable.

Deployment scenarios emerge from the evaluation based on resource constraints and accuracy requirements. ShareDAB-6-6-512 provides near-baseline accuracy (99% retention) with substantial parameter reduction (56% transformer), suitable for production systems prioritizing accuracy. ShareDAB-2-2-256 and ShareDAB-3-2-256 achieve 95% accuracy retention with moderate parameter reduction (47-54% transformer), balancing efficiency and accuracy for general deployment. ShareDAB-3-3-256 and ShareDAB-6-6-256 achieve aggressive compression (67-84% transformer reduction) with accuracy trade-offs (92-94% retention), viable for severely resource-constrained scenarios.

Table 2: ShareDAB-DETR configurations on COCO 2017 val. All models use ResNet-50 backbone (800×1200 input, 50 epochs).

Configuration	Sharing (Enc/Dec)	Total Params (M)	Transformer Params (M)	AP	AP Retention	Param Reduction	Transformer Reduction	Memory (MB)
DAB-DETR (Liu et al., 2022) (Baseline)	–	44.0	19.67	42.2	100%	–	–	6527
ShareDAB-2-2-256	2/2	34.3	9.63	40.1	95.0%	22.0%	51.0%	2234
ShareDAB-3-2-256	3/2	33.0	8.32	40.1	95.0%	25.0%	57.7%	2214
ShareDAB-3-3-256	3/3	31.1	6.42	39.8	94.3%	29.3%	67.3%	2188
ShareDAB-6-6-256	6/6	27.9	3.21	39.2	92.9%	36.6%	83.7%	2164
ShareDAB-6-6-512	**6/6**	**35.8**	**8.65**	**41.8**	**99.0%**	**18.6%**	**56.0%**	**2503**

Table 3: Detailed COCO 2017 metrics and inference efficiency. AP and AP variants computed on val set. Latency and throughput measured on single NVIDIA RTX 3070 GPU with 800×1200 pixel inputs.

Configuration	Sharing (Enc/Dec)	AP	AP50	AP75	APS	APM	APL	Latency (ms)	Throughput (imgs/s)
DAB-DETR (Liu et al., 2022) (Baseline)	–	42.2	63.1	44.7	21.5	45.7	60.3	68.18	14.67
ShareDAB-2-2-256	2/2	40.1	61.9	42.2	21.2	43.5	58.3	74.06	13.50
ShareDAB-3-2-256	3/2	40.1	62.2	42.3	21.2	43.4	58.6	72.77	13.74
ShareDAB-3-3-256	3/3	39.8	61.4	42.1	20.2	43.3	58.4	75.13	13.31
ShareDAB-6-6-256	6/6	39.2	60.4	41.5	21.8	42.7	56.2	71.67	13.95
ShareDAB-6-6-512	**6/6**	**41.8**	**62.8**	**44.5**	**22.0**	**45.3**	**60.1**	**89.75**	**11.14**

5.5 Positioning within DETR Architectures

Table 1 contextualizes ShareDAB-DETR within recent DETR advances. Methods like DN-DETR (48.6 AP) and DINO (49.4 AP) achieve higher accuracy through training innovations, yet maintain 40-44M parameters without addressing memory. ShareDAB-DETR trades accuracy (41.8 AP) for memory efficiency: 2503 MB versus DAB-DETR's 6527 MB enables 4GB GPU deployment. This represents a deliberate design choice for resource-limited scenarios rather than peak accuracy optimization. Notably, CentralBank is complementary to denoising and matching strategies combining our architectural compression with DN-DETR's training innovations could achieve both high accuracy and memory efficiency on edge devices and it can be combined with traditional compression methods such as quantization, pruning, or knowledge distillation.

5.6 Architectural Generalizability and Evaluation Rationale

While this study utilizes DAB-DETR to validate the CentralBank mechanism, the underlying design principles are inherently applicable across the broader spectrum of transformer-based detectors. This section discusses the architectural universality of the approach and clarifies the rationale behind the selected evaluation baseline.

5.6.1 Model-Agnostic Design Principles

CentralBank targets the transformer projection layer, which constitutes a universal architectural bottleneck across all attention-based vision models. By replacing standard linear layers with dynamic wrappers, our mechanism preserves the original attention mechanics, feed-forward computation, and network topology. This design ensures that the implementation is strictly model-agnostic, allowing the same parameter-sharing architecture to be applied without modification to variants such as DN-DETR, DINO, or Deformable DETR. The consolidation of projection matrices into a shared table with deterministic offset mapping achieves compression uniformly, independent of the specific attention formulation used by the detector.

5.6.2 Rationale for DAB-DETR Selection

The selection of DAB-DETR as the primary evaluation platform is motivated by distinct architectural and practical factors. First, its training memory requirement of 6.5 GB (with a ResNet-50 backbone) exemplifies the resource constraints encountered on edge hardware, such as the NVIDIA Jetson Nano. This high baseline consumption ensures that the demonstrated 67% memory reduction translates into a tangible deployment advantage. Second, the 4D dynamic anchor mechanism in DAB-DETR represents a sophisticated query design that is more complex than earlier DETR variants. Validating parameter sharing on this architecture suggests robustness across diverse query formulations. Finally, DAB-DETR occupies a critical middle ground in the accuracy frontier. While newer models achieve higher performance through

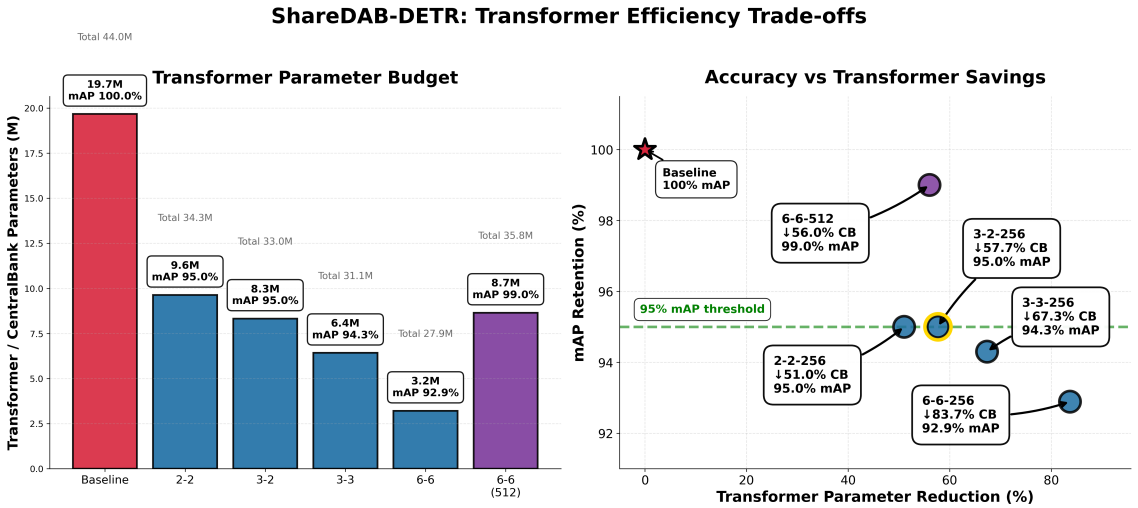

Figure 3: Parameter-accuracy trade-offs for ShareDAB-DETR variants. **Left**: Transformer parameter reduction from baseline 19.67M to 3.21M across configurations. ShareDAB-6-6-512 maintains balanced compression with 8.65M parameters. **Right**: Accuracy retention versus parameter reduction percentage. ShareDAB-6-6-512 achieves near-baseline accuracy with substantial compression.

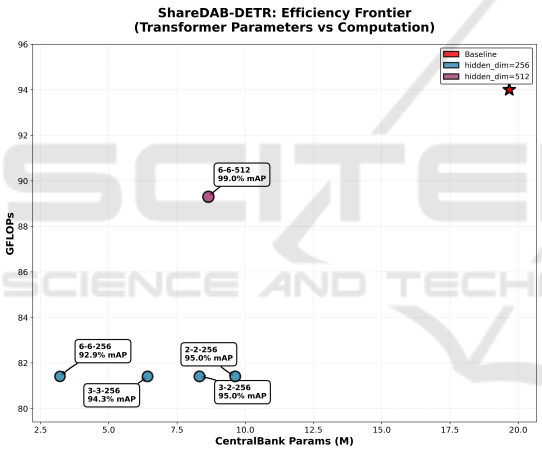

Figure 4: Efficiency frontier: transformer parameters versus GFLOPs versus accuracy. ShareDAB-6-6-512 achieves 8.65M parameters, 89.3 GFLOPs, and 99% AP retention. Hidden dimension variations (512 vs. 256) produce distinct efficiency curves, with larger dimensions demonstrating superior accuracy retention across equivalent sharing levels.

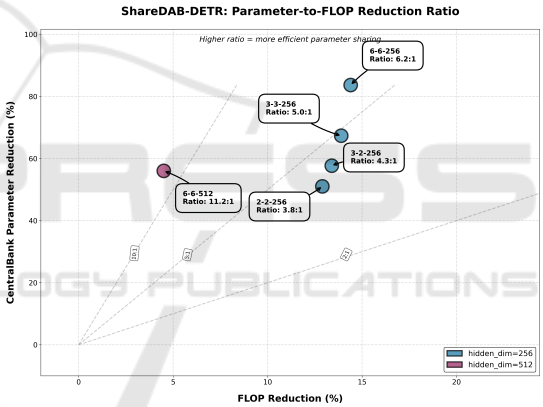

Figure 5: Parameter-to-FLOP efficiency ratios across configurations. Ratios reflect the interaction between parameter sharing, hidden dimension, and backbone dominance. Diagonal reference lines indicate constant efficiency frontiers (2:1, 5:1, 10:1).

advanced training strategies, they share similar parameter profiles. This makes DAB-DETR an ideal candidate for isolating the effects of architectural compression from the influence of training-specific innovations.

5.6.3 Applicability to Broader DETR Architectures

The fundamental compression mechanism is expected to deliver comparable parameter and memory reductions across the DETR family. Although different training dynamics in models like DINO or DN-DETR may influence the specific accuracy retention rates, architectural savings remain constant. Furthermore, the efficiency gains of CentralBank are likely to be amplified in detectors utilizing Vision Transformer backbones. In such architectures, transformer operations frequently account for more than 50% of the total FLOPs, significantly higher than the 29% observed in ResNet-based models. Consequently, applying CentralBank in these contexts would likely yield substantial computational reductions in addition to the established memory savings. Future work will prioritize extending this evaluation to these computation-heavy architectures, as well as lightweight backbones, where relative efficiency gains are most critical for

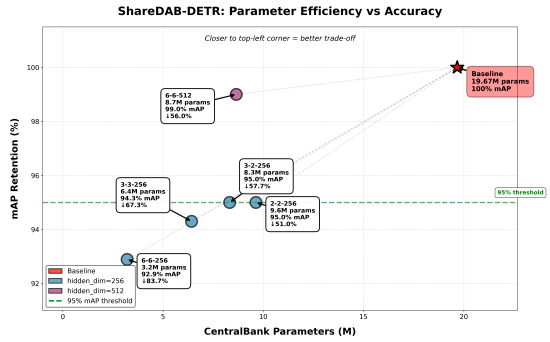

Figure 6: Parameter-accuracy trade-off assessment across deployment scenarios. ShareDAB-6-6-512 achieves 8.65M transformer parameters with 99% AP retention. The 95% AP retention threshold (green line) separates practical and extreme compression regimes. Diagonal reference lines represent Pareto efficiency frontiers.

real-time edge deployment.

6 DISCUSSION

CentralBank primarily targets memory efficiency rather than uniform reductions across all resource dimensions. Although FLOP reductions remain modest (5 - 13%), memory usage is reduced by 67%, reflecting the fact that parameter sharing operates on transformer weights where memory consumption concentrates, while computation remains largely dominated by the backbone. As a result, CentralBank addresses the main feasibility constraint for deployment while offering limited but consistent computational savings.

Our results demonstrate CentralBank's effectiveness as an architectural solution for efficient object detection. Three key relationships emerge from the evaluation:

Hidden dimension trade-offs shape compression-accuracy dynamics. The observed accuracy patterns reveal an important principle: compression-induced capacity loss can be offset through increased representation dimensionality. Since both 6-6-256 and 6-6-512 configurations employ identical layer sharing, the performance difference (41.8 vs. 39.2 AP) arises solely from hidden dimension variation. This finding indicates that optimal sharing configurations must jointly optimize grouping granularity and dimensional capacity rather than treating them as independent design parameters.

View-based slicing decouples memory consumption from access patterns. The consistent memory reduction across all configurations (62-67% from baseline) reflects a fundamental property of CentralBank's architecture: memory consumption depends solely on consolidated parameter table size rather than the number of layers accessing shared parameters. Consequently, configurations with identical hidden dimensions but different sharing levels exhibit comparable memory usage despite substantial parameter count differences. This decoupling makes CentralBank particularly attractive for memory-constrained deployment: developers can prioritize accuracy through conservative sharing while still achieving dramatic memory savings.

ResNet-50 backbone dominance constrains FLOP reduction potential. The modest 5-13% FLOP reduction across configurations reflects the computational distribution in object detection architectures. ResNet-50 backbone accounts for approximately 67 of 94 GFLOPs (Liu et al., 2022) in the baseline configuration. Since CentralBank targets transformer parameters where memory bottlenecks occur, but architectural design places computation primarily in the backbone, computational savings remain limited by this structural imbalance. This finding indicates that CentralBank would provide greater relative computational gains for architectures with larger transformer-to-backbone ratios, such as Vision Transformer-based detectors where transformer operations dominate computational cost.

7 CONCLUSION

We introduced CentralBank, a parameter sharing mechanism that consolidates transformer projections into a unified memory-efficient table accessed through deterministic offset mapping. Applied to DAB-DETR, CentralBank achieves up to 84% transformer parameter reduction and 67% memory reduction while retaining 93-99% of baseline accuracy. Our best configuration (ShareDAB-6-6-512) achieves 41.8 AP with only 8.65M transformer parameters and 2503 MB training memory-a 99% accuracy retention rate that enables practical edge deployment on 4 GB devices.

Comprehensive evaluation across five configurations reveals three key insights: (1) larger hidden dimensions enable aggressive sharing without accuracy loss; (2) memory reduction remains consistent across sharing configurations due to view-based slicing; (3) a 95% AP threshold delineates practical deployment configurations. CentralBank's model-agnostic design and architectural integration distinguish it from post-training compression, offering a principled approach to efficient transformer design.

Our work demonstrates that sharing of architectural parameters can effectively bridge the gap between transformer performance and edge deployment

constraints, enabling powerful object detection on devices with limited resources.

8 FUTURE WORK

Future work will focus on consolidating CentralBank as a general architectural principle. First, we will evaluate its applicability to other vision tasks, such as instance and panoptic segmentation, and to transformer-heavy architectures, particularly Vision Transformer-based detectors. These settings are expected to exhibit larger computational gains than the ResNet-50 baseline, where transformer operations represent a limited fraction of total FLOPs.

Second, we plan a systematic study across other DETR variants, including DN-DETR and DINO, to assess the interaction between parameter sharing and different training dynamics. Extending this analysis to lightweight backbones will further clarify how CentralBank scales across the efficiency spectrum.

Third, we will explore dynamic offset mapping strategies that adapt sharing patterns at runtime based on device-specific memory and compute constraints, enabling hardware-aware optimization.

Finally, we will study the compatibility of CentralBank with complementary compression methods, including quantization, structured pruning, and knowledge distillation, to determine whether their efficiency gains are additive or interacting in non-trivial ways.

ACKNOWLEDGEMENTS

This work has received funding from the European Union's Horizon 2020 programme dAIEDGE (G.A. No 101120726).

REFERENCES

Bae, S., Fisch, A., Harutyunyan, H., Ji, Z., Kim, S., and Schuster, T. (2024). Relaxed recursive transformers: Effective parameter sharing with layer-wise lora. *arXiv preprint arXiv:2410.20672*.

Bolya, D., Fu, C.-Y., Dai, X., Zhang, P., Feichtenhofer, C., and Hoffman, J. (2022). Token merging: Your vit but faster. *arXiv preprint arXiv:2210.09461*.

Brown, N., Williamson, A., Anderson, T., and Lawrence, L. (2023). Efficient transformer knowledge distillation: A performance review. *arXiv preprint arXiv:2311.13657*.

Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., and Zagoruyko, S. (2020). End-to-end object detection with transformers. *arXiv preprint*.

Chang, C.-C., Sung, Y.-Y., Yu, S., Huang, N.-C., Marculescu, D., and Wu, K.-C. (2024). Flora: Fine-grained low-rank architecture search for vision transformer. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 2482–2491.

Chen, Q., Chen, X., Wang, J., Zhang, S., Yao, K., Feng, H., Han, J., Ding, E., Zeng, G., and Wang, J. (2023). Group detr: Fast detr training with group-wise one-to-many assignment. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6633–6642.

Dong, W., Zhang, X., Chen, B., Yan, D., Lin, Z., Yan, Q., Wang, P., and Yang, Y. (2024). Low-rank rescaled vision transformer fine-tuning: A residual design approach. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16101–16110.

Frankle, J., Dziugaite, G. K., Roy, D. M., and Carbin, M. (2019). Stabilizing the lottery ticket hypothesis. *arXiv preprint arXiv:1903.01611*.

Habib, G., Saleem, T. J., and Lall, B. (2024). Knowledge distillation in vision transformers: A critical review. *arXiv preprint*.

Han, S., Mao, H., and Dally, W. J. (2016). Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint*.

Hay, T. D. and Wolf, L. (2024). Dynamic layer tying for parameter-efficient transformers. *arXiv preprint arXiv:2401.12819*.

Hu, J. C., Cavicchioli, R., Berardinelli, G., and Capotondi, A. (2024). Sharebert: Embeddings are capable of learning hidden layers. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38:18225–18233.

Jia, D., Yuan, Y., He, H., Wu, X., Yu, H., Lin, W., Sun, L., Zhang, C., and Hu, H. (2023). DETRs with hybrid matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 19702–19712.

Kang, B. J., Kim, N., and Kim, H. (2025). LRA-QViT: Integrating low-rank approximation and quantization for robust and efficient vision transformers. In Singh, A., Fazel, M., Hsu, D., Lacoste-Julien, S., Berkenkamp, F., Maharaj, T., Wagstaff, K., and Zhu, J., editors, *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 28943–28958. PMLR.

Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., and Soricut, R. (2020). Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint*.

Li, F., Zhang, H., Liu, S., Guo, J., Ni, L. M., and Zhang, L. (2022). DN-DETR: Accelerate DETR training by introducing query DeNoising. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13619–13627.

Li, Z., Xiao, J., Yang, L., and Gu, Q. (2023). Repq-vit: Scale reparameterization for post-training quantization of vision transformers. In *Proceedings of the*

IEEE/CVF International Conference on Computer Vision, pages 17227–17236.

Lin, Y., Zhang, T., Sun, P., Li, Z., and Zhou, S. (2023). Fq-vit: Post-training quantization for fully quantized vision transformer. *arXiv preprint*.

Liu, D., Zhu, Y., Liu, Z., Liu, Y., Han, C., Tian, J., Li, R., and Yi, W. (2025). A survey of model compression techniques: past, present, and future. *Frontiers in Robotics and AI*, 12.

Liu, H. (2022). Exploration of knowledge distillation methods on transformer language models for sentiment analysis. Master's thesis, KTH Royal Institute of Technology, Stockholm, Sweden.

Liu, R., Yang, K., Roitberg, A., Zhang, J., Peng, K., Liu, H., Wang, Y., and Stiefelhagen, R. (2024). Transkd: Transformer knowledge distillation for efficient semantic segmentation. *IEEE Transactions on Intelligent Transportation Systems*.

Liu, S., Li, F., Zhang, H., Yang, X., Qi, X., Su, H., Zhu, J., and Zhang, L. (2022). Dab-detr: Dynamic anchor boxes are better queries for detr. *arXiv preprint*.

Lu, C., Zhang, J., Chu, Y., Chen, Z., Zhou, J., Wu, F., Chen, H., and Yang, H. (2022). Knowledge distillation of transformer-based language models revisited. *arXiv preprint arXiv:2206.14366*.

Mehta, S. and Rastegari, M. (2021). Mobilevit: lightweight, general-purpose, and mobile-friendly vision transformer. *arXiv preprint arXiv:2110.02178*.

Meng, D., Chen, X., Fan, Z., Zeng, G., Li, H., Yuan, Y., Sun, L., and Wang, J. (2021). Conditional DETR for fast training convergence. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 3651–3660.

Moon, J., Kim, D., Cheon, J., and Ham, B. (2024). Instance-aware group quantization for vision transformers. *arXiv preprint*.

Sanh, V., Debut, L., Chaumond, J., and Wolf, T. (2020). Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint*.

Shao, W., Chen, M., Zhang, Z., Xu, P., Zhao, L., Li, Z., Zhang, K., Gao, P., Qiao, Y., and Luo, P. (2024). Omniquant: Omnidirectionally calibrated quantization for large language models. *arXiv preprint*.

Shi, D., Tao, C., Jin, Y., Yang, Z., Yuan, C., and Wang, J. (2023). Upop: Unified and progressive pruning for compressing vision-language transformers. In *International Conference on Machine Learning*, pages 31292–31311. PMLR.

Tang, Y., Han, K., Wang, Y., Xu, C., Guo, J., Xu, C., and Tao, D. (2022). Patch slimming for efficient vision transformers. *arXiv preprint*.

Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A., and Jégou, H. (2021). Training data-efficient image transformers & distillation through attention. *arXiv preprint*.

Wang, Y., Zhang, X., Yang, T., and Sun, J. (2022). Anchor detr: Query design for transformer-based detector. In *Proceedings of the AAAI conference on artificial intelligence*, volume 36, pages 2567–2575.

Wu, Z., Chen, J., Zhong, H., Huang, D., and Wang, Y. (2024). Adalog: Post-training quantization for vision transformers with adaptive logarithm quantizer. In *European Conference on Computer Vision*, pages 411–427. Springer.

Wu, Z., Wang, S., Zhang, J., Chen, J., and Wang, Y. (2025). Fima-q: Post-training quantization for vision transformers by fisher information matrix approximation. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 14891–14900.

Xie, K., Lu, S., Wang, M., and Wang, Z. (2021). Elbert: Fast albert with confidence-window based early exit. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7713–7717. IEEE.

Xu, L., Pang, S., Qiu, W., Wu, Z., Bai, X., Mei, K., and Xue, J. (2024a). Redundant queries in detr-based 3d detection methods: Unnecessary and prunable. *arXiv preprint arXiv:2412.02054*.

Xu, T. et al. (2024b). A survey on transformer compression. *arXiv preprint arXiv:2402.05964*.

Yu, L. and Xiang, W. (2023). X-pruner: explainable pruning for vision transformers. *arXiv preprint*.

Yu, S., Chen, T., Shen, J., Yuan, H., Tan, J., Yang, S., Liu, J., and Wang, Z. (2022). Unified visual transformer compression. *arXiv preprint arXiv:2203.08243*.

Yuan, Z., Xue, C., Chen, Y., Wu, Q., and Sun, G. (2024). Ptq4vit: Post-training quantization for vision transformers with twin uniform quantization. *arXiv preprint*.

Zhang, H., Li, F., Liu, S., Zhang, L., Su, H., Zhu, J., Ni, L. M., and Shum, H.-Y. (2023). DINO: DETR with improved DeNoising anchor boxes for end-to-end object detection. In *International Conference on Learning Representations (ICLR)*.

Zheng, C., Zhang, K., Yang, Z., Tan, W., Xiao, J., Ren, Y., Pu, S., et al. (2022). Savit: Structure-aware vision transformer pruning via collaborative optimization. *Advances in Neural Information Processing Systems*, 35:9010–9023.

Zhong, Y., Huang, Y., Hu, J., Zhang, Y., and Ji, R. (2025). Towards accurate post-training quantization of vision transformers via error reduction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Zhu, M., Tang, Y., and Han, K. (2021a). Vision transformer pruning. *arXiv preprint*.

Zhu, X., Su, W., Lu, L., Li, B., Wang, X., and Dai, J. (2021b). Deformable detr: Deformable transformers for end-to-end object detection. *arXiv preprint*.