DEGREE OF DOCTOR OF PHILOSOPHY IN
COMPUTER ENGINEERING AND SCIENCE

DOCTORATE SCHOOL IN
INFORMATION AND COMMUNICATION TECHNOLOGIES

XXVIII CYCLE

UNIVERSITY OF MODENA AND REGGIO EMILIA

"ENZO FERRARI" ENGINEERING DEPARTMENT

Ph.D. DISSERTATION

Loosely Schema-aware Techniques for Big Data Integration

Candidate:
Giovanni SIMONINI
Advisor:
Prof. Sonia BERGAMASCHI
Co-Advisor:
Prof. H.V. JAGADISH
Director of the School:
Prof. Giorgio Matteo VITETTA

DOTTORATO DI RICERCA IN
COMPUTER ENGINEERING AND SCIENCE

SCUOLA DI DOTTORATO IN
INFORMATION AND COMMUNICATION TECHNOLOGIES

XXVIII Ciclo

UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

DIPARTIMENTO DI INGEGNERIA "ENZO FERRARI"

TESI PER IL CONSEGUIMENTO DEL TITOLO DI DOTTORE DI RICERCA

Tecniche di Approssimazione di Schema per Integrazione di Grandi Quantitativi di Dati

Tesi di:
Giovanni SIMONINI
Relatore:
Prof. Sonia BERGAMASCHI
Co-Relatore:
Prof. H.V. JAGADISH
Il Direttore della Scuola:
Prof. Giorgio Matteo VITETTA

Keywords:

Data Integration
Big Data
Entity Resolution
Blocking
Topic Detection

Abstract

A huge amount of (semi-)structured data is available on the Web in the form of web tables, marked-up contents (e.g. RDFa, Microdata), and Linked Open Data. For enterprises, government agencies, and researcher of large scientific projects, this data can be even more valuable if integrated with their proprietary data, which are typically subject of traditional Data Integration processes. Being able (i) to retrieve useful data sets and (ii) to identify records that refer to the same entity are fundamental steps to make sense of this data.

The first task may be tackled by identifying the topic of the data sources with respect to a known vocabulary, hence enabling search engines to efficaciously classify them. The second task is a well known problem, called Entity Resolution (ER). Generally, to perform ER and vocabulary-based topic detection, traditional techniques are based on schema-alignment among data sources (i.e., deriving a unique homogenous common schema from several heterogeneous ones). Unfortunately, the (semi-)structured data of the Web is usually characterized by high heterogeneity, volume and noise (missing/inconsistent data), making schema-alignment techniques no longer applicable. Therefore, Data Integration techniques dealing with this type of data typically renounce to exploit data source schemata.

This dissertation presents a set of novel techniques to induce *loose* schema information directly from the data, without exploiting the semantic of the schemas, able to scale to the huge data of the Web. This lose schema information can be employed as a surrogate of the schema-alignment and employed to enhance ER and vocabulary-based topic detection. For ER, I present BLAST (Blocking with Loosely-Aware Schema Techniques), an approach to reduce the ER complexity with indexing techniques aiming to group similar records in *blocks*, and limit the comparison to only those records appearing in the same block. For the topic detection, I propose WHATSIT a novel approach that generates signatures of sources that are matched against the signatures of a reference vocabulary. Thus, a description of the topics of the source in terms of this reference vocabulary is generated. Finally, I developed a software prototype for both the approaches and I experimentally evaluated them on real world datasets. The results demonstrate that BLAST

can outperform the state-of-the-art blocking approaches, and that WHATSIT can actually be employed to detect topics of a given data source.

Sommario

Un grande quantitativo di dati semi-strutturati è disponibile sul Web in forma di tabelle, contenuti annotati (e.g. RDFa, Microdata), e Linked Open Data. Per le aziende, le agenzie governative e i ricercatori di grandi progetti scientifici, questi dati possono costituire una preziosa risorsa se integrati con i dati che già possiedono e che tipicamente sono già oggetto di tradizionali processi di integrazione. Essere in grado di (i) trovare data set utili e (ii) identificare record che si riferiscono alla stessa entit del mondo reale sono processi fondamentali per sfruttare correttamente questi dati.

Il primo task pu essere raggiunto identificando i topic di una sorgente dati con l'ausilio di un vocabolario di concetti noti, permettendo ai motori di ricerca di classificare in maniera corretta ed efficace queste sorgenti. Il secondo task un problema noto, ed è conosciuto come Entity Resolution (ER). Generalmente, per effettuare ER e topic detection basata su un vocabolario, le tecniche tradizionali si basano sull'allineamento dello schema delle sorgenti dati. Sfortunatamente, i dati (semi-)strutturati del Web sono tipicamente caratterizzati da alta eterogeneità, volume e rumore (dati mancanti o inconsistenti), rendendo l'allineamento degli schemi non raggiungibile. Pertanto, le tecniche impiegate per l'integrazione di questo tipo di dati sono tipicamente rinunciano a sfruttare l'informazione relativa agli schemi delle sorgenti dati.

Questa dissertazione presenta un set di nuove tecniche per indurre *informazione approssimata dello schema* direttamente dai dati, senza sfruttare la semantica degli schemi e che scalino con i grossi quantitativi di dati tipici del Web. Questa informazione approssimata può essere impiegata come surrogato dell'allineamento dello schema per migliorare l'ER a il topic detection basata su un vocabolario. Per l'ER, qui presento BLAST Blocking with Loosely-Aware Schema Techniques, un approccio per ridurre la complessità dell'ER con tecniche di indicizzazione che mirano a raggruppare i record in blocchi di record simili e a confrontare solamente quelli che compaiono in uno stesso blocco assieme. Per il topic detection, propongo WHATSIT un nuovo approccio per generare *signatures* di una sorgente dati che poi vengono comparate con le signatures di un vocabolario di riferimento. In questo modo una descrizione della sorgente (i topics) viene fornita con riferimento al vocabolario utilizzato. Infine, ho sviluppato un prototipo dei due approcci proposti ed effettuato esperimenti su data set reali. I risultati dimostrano che BLAST

può raggiungere risultati migliori rispetto allo stato dell'arte per quanto riguarda le tecniche di blocking; mentre WHATSIT può effettivamente essere impiegato per effettuare topic detection basata su un vocabolario di topic di riferimento.

“There are no facts, only interpretations.”

Friedrich Nietzsche

Acknowledgments

Acknowledgments
I owe my deepest gratitude to both my advisor Professors Sonia Bergamaschi, and co-advisor H.V. Jagadish, for their precious support and guide during my Ph.D.

Ringraziamenti
Ringrazio tutti quelli che mi sono stati vicini durante questi tre anni. Ringrazio mia madre, che mi ha sempre sostenuto e permesso di raggiungere i miei obiettivi. Ringrazio la mia ragazza Elena, per essere stata sempre presente e per avermi sostenuto. Grazie anche ai suoi genitori che mi hanno sempre ben accolto (e ben nutrito). Ringrazio mio padre e la cara nonna che ci ha lasciati da poco. Un grazie a tutti i componenti del DBGroup, in particolare a Fabio, compagno di banco e di questo percorso. Ringrazio anche tutti gli amici di Castelvetro e della compagnia del Seme. Ringrazio infine Chiara e Giuseppe che hanno reso il mio soggiorno in America molto piacevole e un po' meno cinese di quanto sarebbe altrimenti stato.

Contents

List of Figures

List of Tables

List of Definitions

List of Examples

Chapter 1

Introduction

The Web has become a valuable source of structured and semi-structured data. A huge amount of high-quality relational data can be extracted from HTML tables [Caf+08; Qiu+15; CP11]; an increasing amount of websites have started to semantically markup their contents (e.g. products, people, and places) using Microformats, Microdata and RDFa [MPB14]; finally, with the advent of the Web of Data, the amount of semi-structured data publicly available as Linked Data is exponentially growing [BHBL09].

The true potential of this data is expressed when different sources are integrated, as demonstrated by recent efforts in mining the web to extract entities, relationships, and ontologies to build large-scale general purpose knowledge bases, such as Freebase[1] and Yago[2] [DS15]. For enterprises, government agencies, and researchers of large scientific projects, this data can be even more valuable if integrated with the data that they already own, and that are typically subject of traditional data integration processes. In this context, data integration is a complex process due to the volume, high heterogeneity and noise of the involved data [DS15], and adapting traditional techniques is not always feasible.

In fact, one fundamental step in data integration is *Entity Resolution* (ER), namely the task of matching records from several data sources that refer to the same real-world entity [Chr12a]; and traditional ER techniques typically rely on the achievement of a schema-alignment among the data sources. Unfortunately, in

[1] http://www.freebase.com/

[2] http://www.mpi-inf.mpg.de/departments/databases-and-information-systems/research/yago-naga/yago//

the context of the *big data* of the web, schema-alignment is not always achievable [DS15; Pap+13]. For instance, Google Base contains over 10,000 entity types that are described with 100,000 unique schemata; and, in such a scenario, performing and maintaining a schema alignment is impracticable [Mad+07].

Moreover, being able to retrieve the right data source and understand its content is a fundamental requirement to the exploitation of the large data source of the web. The content of a data source is better realised when seen in relationship to some reference vocabularies that are already known and understood; but, to compare the data sources to a reference vocabulary, a schema alignment must be achieved, and this (again) is not always possible in the context of the highly heterogenous and noisy data of the web.

In this thesis, I propose novel techniques to extract *loose schema information*, i.e., statistics extracted directly from the data that can be used to approximately describe the schema of a data source. This information can be employed as surrogate of the schema alignment in many scenarios, while its extraction is proven to be scalable to the huge and heterogeneous data of the web. In particular, I propose two novel approaches based on the extraction of loose schema information: the first, to support ER; the second, to support *vocabulary-based topic detection* of data sources (a novel approach to get an insight of a data source presented here). These approaches are introduced in the following sections 1.1 and 1.2 respectively.

1.1 Entity Resolution and Blocking

Entity Resolution (ER) is the task of clustering entity profiles that refer to the same real-world entity. Comparing all possible pairs of profiles of an entity collection is inherently a quadratic problem: if the number of entity profiles grows linearly, then the number of possible comparison grows quadratically. Therefore, a *brute-force* approach becomes infeasible for very large datasets. For this reason indexing techniques are widely employed to group similar profiles into *blocks*, and execute the comparisons only between those appearing in the same block.

Traditional *blocking* techniques generate blocks according to a blocking criterion (*blocking key*), either based on a single attribute, or a combination of attributes

[Chr12a]. So, if two datasets have a different schema, before performing the ER, a schema alignment between the data sources must be achieved. Unfortunately, the semi-structured data of the Web is typically characterized by high heterogeneity, high levels of noise (missing/inconsistent data), and very large volume, making traditional schema alignment techniques no longer applicable [Mad+07; Pap+13]. To solve this problem, *schema-agnostic* blocking approaches have been proposed [Pap+13; MT13a; MT13b; PPK14]. These approaches rely on redundancy to achieve high recall (i.e., the percentage of detected duplicate): each profile is placed in multiple blocks, which significantly reduce the likelihood of missing matches. For instance, in Token Blocking [Pap+13] each token appearing in the dataset values is a blocking key. Thus, each block is associated to a token and contains all the profiles in which that token appears, as illustrated by the example in Fig 1.1 a-b. The downside of schema-agnostic, redundancy-based blocking is the degradation of efficiency. In fact, to ensure a higher recall, low efficient blocks are produced, i.e., a high number of non-matching profiles are placed in the same blocks. To overcome this issue, *meta-blocking* approaches have been proposed [Pap+14; PPK14].

Meta-blocking is the task of restructuring a set of blocks to retain only most promising comparisons. *Unsupervised graph-based* meta-blocking represents a block collection as a weighted graph, called *blocking graph* (example in Figure 1.1c), where each entity profile is a node and an edge exists between two nodes if the corresponding profiles appear at least in one block together. The edges are weighted to capture the likelihood of a match (e.g., in Figure 1.1c the weight is the number of co-occurrence of profiles in the blocks) and edge pruning thresholds are applied to retain only those more promising (Figure 1.1d). At the end of the process, each pair of nodes connected by an edge forms a new block. Differently, *supervised graph-based* meta-blocking associates to each edge a vector of schema-agnostic features (e.g. graph topological measures), and treats the problem of identifying promising edge as a *classification* problem.

My Contribution: I observe that existing meta-blocking techniques leverage exclusively on schema-agnostic features extracted from the target blocking collection. Therefore, inspired by the *attribute-match induction* approaches [Pap+13;

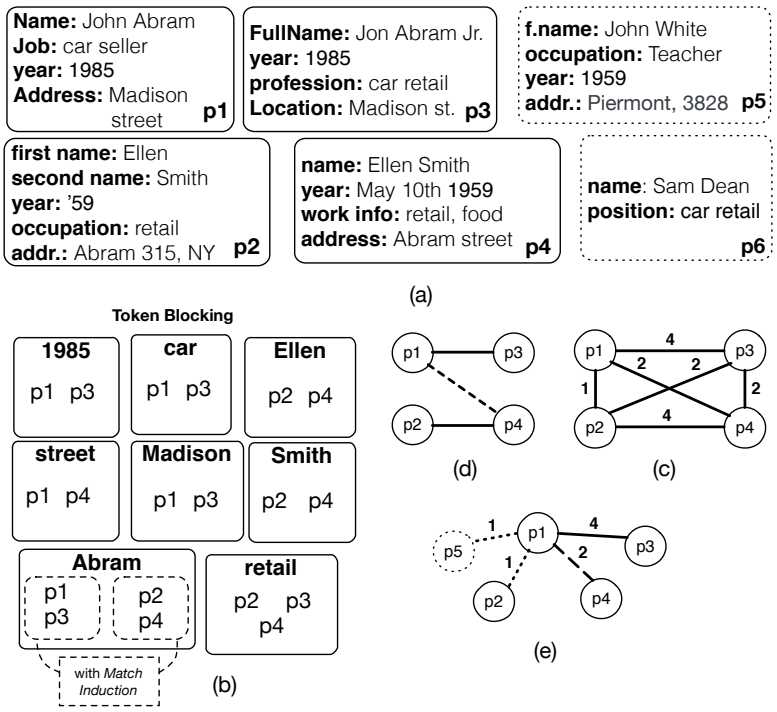


Figure 1.1: Example of *Meta-blocking* processing.

MT13b], able to exploit schema information extracted directly from the data to enhance the quality of the blocks, I argue that a holistic approach combining meta-blocking and *loosely schema-aware* techniques should be attempted. In fact, I notice that, even for highly heterogeneous and voluminous datasets, I can afford to relax the condition of complete *schema-agnosticism* by collecting significant statistics (e.g. co-occurrences and entropies of values in the attributes) that approximately describe the data sources schemas. This *loose* schema information can be exploited during both the blocking and meta-blocking phases to produce high quality blocking collections. The intuition is that some attributes are more informative than others, therefore Shannon entropy can be employed to capture this characteristic. For instance, if I consider independent datasets containing thousands of records about people personal information (as in Figure 1.1), the attribute like *year of birth* or *job position* are generally less informative than the attribute like *name*, because the number of distinct values of the attribute *year of birth* is typically lower than that of the attribute *name*.

In this thesis, I introduce BLAST (Blocking with Loosely-Aware Schema Techniques), a blocking approach suited for the highly heterogeneous and noisy data of the Web. BLAST merges *loosely schema-aware* blocking techniques to an unsupervised meta-blocking approach able to exploit loose schema information to produce high quality blocks. I introduced this approach for the first time in [SBJa; SBJb].

Here the list of contributions introduced with BLAST:

- I propose an approach to extract *loose schema information* from the datasets based on an attribute-match induction technique;
- I present a LSH-based support for attribute-match induction that enables to efficiently scale this latter when required, i.e., with high-dimensional datasets;
- I present an unsupervised graph-based meta-blocking approach able to leverage on loose schema information extracted during the attribute-match induction phase;
- I evaluate my approach on real-world datasets, comparing it against the unsupervised meta-blocking state of the art approaches; moreover, I show how my unsupervised approach can outperform in many case also the supervised meta-blocking state of the art approaches.

1.2 Identifying Data Source Topics

Traditional search engines are designed to operate only on document content, which means that the data of these structured data sources are left out of their exploitation sphere.

Identifying the topics of the online structured data sources is of paramount importance since it will allow them to be indexed by search angines and references to their content be included in query answers. Indexing structured data is fundamentally different and significantly more challenging than indexing flat, unstructured documents, since the structure plays an important role in the semantics of each value in the data. To overcome this limitation, structured data portals like CKAN[3]

[3] http://ckan.org

have been developed. They play a role similar to the one search engines play for web documents, but are based on metdata information that has been explicitly provided by the owners of the datasets. Providing this meta-data information is a tedious and error prone task that can also introduce bias. Furthemore, it is an approach that does not scale easily. Thus, there is a need for a way to identify the topics of the sources in a way that is automatic, can scale at large, and is also robust to the heterogeneity that is typically observed across independently developed data sources.

The ability to automatically identify the topics of the data structured data sources will make them equally important to the static web pages, will promote further the idea of open data, contribute significantly towards the materialization of the "Web of Data" (as opposed to the "Wed of Documents"), and will offer countless opportunities for large scale data analytics [Dha13]. This need has already been recognized and they are not few the efforts to exploit the part of the web that is hidden behind web forms and composable links, i.e., the so-called *hidden web* [Wri08]. These efforts have focused on the part of the web accessed through web forms and not to sources that expose their datasets directly. Furthemore, although there have been efforts to add semantics to the values of the online structured data sources, these works have become restricted to the value-level, ignoring the important semantic information that the structure and the schema in general can offer [Mad+09].

My Contribution: I advocate that it is possible to recognize the concepts used in a data source by exploiting loose schema information, i.e., statistics extracted directly from the source values. In particular, I claim that the entropy of the set of values of an attribute in the data can be used as an identifier of the specific attribute. The advantage of the entropy is that it does not depend on the actual values of the attribute, but on their distributions in a specific domain. This distribution, in turn, does not depend on a specific source, but is a feature of the domain represented by the attribute. All these are making entropy a very promising identifier of what an attribute is describing. The identifiers of the different attributes in the data can be combined together to form a signature of the concepts represented in the source. It is of course possible that two different attributes have the

same entropy value. This means that the entropy is not always a unique identifier for an attribute, but rather a "*loose*" identifier (and classified as *loose* schema information). Even though it is not unique and *precise* signature, I can use this loose signature for identifying concepts, because I can consider the combination of the entropies of the individual attributes in the way that they are modeled in the source, reducing significantly the chances that a concept would be mistakenly taken for another. The concept signatures in a source collectivelly can form a signature of the contents of the source.

The representation of the concepts in a source is definitely not a complete representation. This is a consequence to the heterogeneity that is naturally embedded in every source and highly depends on the data of interest upon which the source has been created. For this reason, and to obtain a more accurate semantic representation of the concepts in the source, I propose to match their signatures against a vocabulary of signatures that is more complete, and use these signatures in the vocabulary as representations of the source. I call this approach *vocabulary-based topic detection*.

The use of statistical properties for recognizing semantically equivalent or related properties has been exploited with success in the past, and in particular in the field of schema matching [KN03]. Inspired by that work, I extend the idea and apply it in the area of source topic detection. Existing approaches have so far been based on the "classical" values of the Shannon entropy. My experiments have shown that these metrics need to be normalized and may generate unstable results in real environments. This is because the frequency distribution of repeated values in real property domains is typically right skewed, i.e. only few values are significantly repeated, which is problematic because the entropy and mutual information are typically sensitive to regions corresponding to small probabilities, and also because their range depends on the cardinality of the attribute domain. This makes the classical values of entropy and mutual information not usable for my case, and I instead use pseudo-additive versions of the classic Shannon entropy.

My vocabulary based topic detection approach has been introduced for the first time in [Ber+14] and extended in [Ber+]. The specific contributions of this work can be summarized as follows:

- I propose a technique for modeling source topics that is independent of the names of the source structures, hence, independent of many of the complications that structural heterogeneity introduces. The technique uses some statistic metric to generate identifiers of the various attributes that combined together form signatures of the concepts mentioned in the data source.
- I illustrate that the traditional entropy measure is very sensitive and not suit for many practical cases, so I introduce and use a pseudo-additive versions of it;
- I use my technique to effectively create a required vocabulary of concept signatures based on the information provided by DBpedia[4];
- I use a matching algorithm to match the generated concept signatures of the source to signatures of concepts in the reference vocabulary;
- I describe the materialization of my theory into a system called WHATSIT; and
- I provide an extensive set of experimental evaluation with real data that illustrates the effectiveness of my approach and discuss my interesting findings.

1.3 Structure of the Thesis

This thesis is structured as follows: chapter 2 reviews the related works; chapter 3 gives the preliminary concepts and definition employed throughout all the thesis; chapter 4 presents the novel techniques to extract the loose schema information from the data sources; chapter 5 and 6 present the novel approaches that employ loose schema information for blocking and vocabulary based topic detection respectively; in chapter 7 the experimental outcomes are presented and discussed; and finally, in chapter 8, I draw the conclusions and present ongoing and future works.

[4] http://dbpedia.org

Chapter 2

Related Work

2.1 Entity Resolution

Blocking techniques have been commonly employed in *Entity Resolution*, [Chr12b; KR10; GM13; NH10; GM12] for a survey, and can be classified into two broad categories [DS15]: the *schema-based* approaches (e.g. Suffix Array [Vri+11], q-grams blocking [Gra+01], Canopy Clustering [MNU00], HARRA [KL10]) that rely on a schema mapping [BLN86; RB01] among the sources to ensure high level of efficiency; and the *schema-agnostic* approaches that are applicable when the schema mapping is not achievable as in the context of the (semi-)structured data of the Web. In the latter category falls: Token Blocking [Pap+13], Total Description [Pap+12], and *Attribute-Matching induction* based techniques [Pap+13; MT13b]. *Attribute Clustering* [Pap+13] relies on the comparison of all possible pairs of attribute profiles of two datasets to find the pairs of those most similar; this is a inefficient process, because the vast majority of comparisons are superfluous, and my LSH-based variation of attribute-match induction aims to address this specific issue. In the same category, *TYPiMatch* [MT13b] tries to identify the latent *subtypes* from generic attributes (e.g. *"description"*, *"info"*, etc.), frequent on generic dataset of the Web, that can be exploited as auxiliary information for both *schema-based* and *schema-agnostic* blocking techniques.

In general, both schema-based and schema-agnostic approaches can produce

redundancy-positive[1] block collections [Pap+13; PPK14]. Meta-blockign approaches [Pap+14; PPK14] aim to reduce the number of the redundant comparisons restructuring the *block collection* produced by the underlying blocking techniques.

If having a training sample of annotated matching pairs is possible, a higher efficiency can be achieved through *supervised meta-blocking* [PPK14]; but the training set is indispensable, therefore it may not be a practical solution in many scenario. For instance, if the training set is not available, it has to be collected by experts of the domain (if the domain is complex), or by crowdsourcing. Both the solutions require time and resources that might not be available, hence the unsupervised meta-blocking remains a valuable alternative; BLAST falls in this category, but, differently from any of the other existing unsupervised approach, it is tailored to exploit information generated with a attribute-match induction technique. Moreover, it can outperform even the supervised meta-blocking state of the art approach in many contexts.

2.2 Insight into Data Source Topics

To provide users with tools for automatically understanding the content of a data source is a hot and challenging task. The problem is well known in the IR Community, and commonly addressed exploiting *topic modeling* techniques [Ble12; WC06] to cluster and retrieve textual documents according to their topics. These approaches are based on the assumption that the same topic can be identified in different documents by means of latent patterns in the text (i.e., relations among words), typical of every language. This assumption does not hold in the context of structured data, since the information is no longer represented as a monolithic document, but instead, as a graph, such as the *Entity-Relationship* model [Che76] and the *RDF* model [2], where the relationships among concepts are explicitly modeled by means of the metadata. In the database and semantic web literature, two main classes of solution have been proposed to automatically support the target users (e.g., data scientists, statisticians, data engineers, etc.) of structured data:

[1] The similarity of two entity profiles is proportional to the number of blocks they share.

[2] http://www.w3.org/TR/WD-rdf-syntax-971002/

Summary-based approaches [YPS09; YPS11; ZCQ07; YJ06], that aim to provide a summary of a target data source; and *Ontology Matching* approaches, that allow to map a known ontology to the one employed to the target data source.

Summary-based approaches aim to identify and extract a small subset of the information which is representative of the entire contents of the data source. In [YPS09] and [YPS11] two approaches dealing with relational databases and graphs, respectively, have been proposed. Both the approaches compute the closeness between data structures and the importance of the data taking into account entropy and mutual information. In [Ber+07], the goal is to summarize an attribute domain. A mix of techniques is applied for clustering the attribute values and identifying in each cluster a single representative value. The limit of these approaches is that the produced summary maintains the same semantic of the original dataset and, therefore, a user must be able to understand such semantic (e.g. names of the classes and properties) to understand the summary itself.

Ontology-based approaches [SE13; Rah11; CSH06; ES13; Sch+12] try to match content and data structures into some reference ontology, and can be generally classified, following [ES13], in: *schema-based* and *instance-based* mapping. The former aims to map ontologies relying on the schema information, e.g., trying to map classes and properties on the basis of their names; while the latter try to align ontologies using their *instances*. The intuition behind the *instance-based* approaches is that when two concepts are associated to the same set of objects (e.g. property names and their values), they are likely to be similar [Sch+12]. Thus, the *instance-based* approaches can overcome the *schema-based* approaches when is difficult to identify the semantic similarities of the elements of the schema [KN03].

I note that my proposal differs from *Ontology Matching* approaches in a fundamental aspect: my goal is not to determinate a fully correct (e.g. identifying class and property hierarchies) and complete match between ontologies; I rather aim to support the identification of some classes of a wide reference ontology (e.g., DBpedia), that could be used to describe the topics of a data source. My approach could be employed to support *instance-based* matching, though; this is an orthogonal problem that I do not tackle in this paper.

In [HFJ12] *Mutual Information* is employed to characterize RDFS graphs cap-

turing the statistical association of classes and properties in an ontology; this information is then exploited to map user terms to the most appropriate element in a *schema-free* querying system. Nevertheless, in this thesis I adopted a novel technique for estimating the mutual information based on likelihood. The idea of creating a data source signature starts from [KN03] where a dependency graph is built for supporting schema matching in a data integration approach. In this thesis I adapted the approach for RDF sources and I extended the technique with the introduction of different kinds of edges connecting nodes. Moreover, this thesis radically modifies my previous proposal [Ber+14], where composite likelihood has been experimented for the same purposes. Deep evaluation showed that a best performance is achieved with the measures here proposed.

Finally, it is important to observe that Sindice.com [Ore+08], an RDF search engine, could be considered as a possible solution of the problem on hand. Nevertheless, Sindice focuses on finding triples containing particular keywords and not discovering data sources topics.

Chapter 3

Preliminaries

This chapter defines preparatory concepts and notation employed throughout the thesis. In particular, section 3.1 gives the basic definitions about blocking techniques for Entity Resolution and presents the problem of *attribute-match induction* and *meta-blocking*, while section 3.2 defines the problem of identifying topics of a data source with respect to a reference vocabulary.

3.1 Blocking

An *entity collection* is a set of profiles. An *entity profile* p is a tuple $\langle id_p, \mathcal{I}_p^{A\varepsilon} \rangle$, where id_p is a unique identifier and $\mathcal{I}_p^{A\varepsilon}$ is a set of *name-value* pairs $\langle a, v \rangle$, instances of the set of *attribute-name* $A_\mathcal{E}$ associated to an entity collection $\mathcal{E}$. Two profiles $p_i, p_j \in \mathcal{E}$ are *matches* ($p_i \approx p_j$) if they refer to the same real world object, and *Entity Resolution* (ER) is the task of identifying those matches given $\mathcal{E}$.

There exist two kind of ER [PPK14]: *clean-clean* ER and *dirty* ER. The former takes as input two duplicate-free entity collections $\mathcal{E}_1$ and $\mathcal{E}_2$ and compares pairs $\{(p_i, p_j) \mid p_i \in \mathcal{E}_1, p_2 \in \mathcal{E}_2\}$; the latter takes as input a single collection $\mathcal{E}_s$ containing duplicates and compares all possible pair of profiles. The naive solutions to clean-clean and dirty ER imply respectively $|\mathcal{E}_1| \times |\mathcal{E}_2|$ and $\binom{|\mathcal{E}_s|}{2}$ comparisons, where $|\mathcal{E}_i|$ is the cardinality of an entity collection $\mathcal{E}_i$. *Blocking* approaches aims to reduce this complexity by indexing similar profiles into *blocks* according to a *blocking key* (i.e., the indexing criterion), restricting the actual comparisons of

profiles to those appearing in the same block. A set of blocks $\mathcal{B}$ is called *blocking collection*, and its *aggregate cardinality* is $\|\mathcal{B}\| = \sum_{b_i \in \mathcal{B}} \|b_i\|$, where $\|b_i\|$ is the number of comparisons implied by the block b_i.

In this thesis I follow best practices to establish the quality of a blocking collection [KL10; Pap+13; Pap+14; PPK14]: the problem of determining if two profiles actually refer to the same real-world object is not a concern of my work; I rather assume that exists an oracle that can determine that. I employ *Pair Completeness* (PC) and *Pair Quality* (PQ) [Chr12a] to evaluate the quality of a blocking collection $\mathcal{B}$, two well known surrogates of *recall* and *precision* respectively. $PC(\mathcal{B})$ measures the portion of duplicate profiles that are placed in at least on block; while $PQ(\mathcal{B})$ measure the the portion of useful comparison, i.e., those that detect a match. Formally:

$$PC(\mathcal{B}) = \frac{|\mathcal{D}^{\mathcal{B}}|}{|\mathcal{D}^{\mathcal{E}}|} \tag{3.1}$$

$$PQ(\mathcal{B}) = \frac{|\mathcal{D}^{\mathcal{B}}|}{\|\mathcal{B}\|} \tag{3.2}$$

where $\mathcal{D}^{\mathcal{B}}$ is the set of duplicates appearing in $\mathcal{B}$ and $\mathcal{D}^{\mathcal{E}}$ is the set of all duplicates in the collection $\mathcal{E}$.

In the context of the highly heterogeneous data of the Web, a feature of the blocking criteria that has been proved to be necessary for ER is *redundancy* [Pap+13]; indeed, if an entity profile can be indexed by more than one blocking key, the chance of missing matches decrease. For instance, Token Blocking [Pap+13] considers each token appearing in the values of an entity collection as a blocking key, independently of the attributes in which it appears. This allows to achieve high PC, but at the expense of PQ.

Two classes of unnecessary comparisons can be distinguished: (*redundant* comparisons), entailing comparison of entity profiles more than once; and (*superfluous* comparisons), entailing comparison of non-matching profiles ($p_i \not\approx p_j$). **Attribute-match induction** approaches can be employed to enhance schema-agnostic blocking limiting the superfluous comparisons. **Meta-blocking** is the state of the art approach to reduce both superfluous and redundant comparisons from an existing blocking collection.

In the following I formally define attribute-matching induction and meta-

blocking.

3.1.1 Attribute-match Induction

The goal of attribute-match induction is to induce groups of "similar" attributes between two collections $\mathcal{E}_1$ and $\mathcal{E}_2$ from the distribution of the attribute values, without exploiting the semantic of the attribute names.

Definition 3.1. ATTRIBUTE-MATCH INDUCTION. Given two entity collections $\mathcal{E}_1, \mathcal{E}_2$, *attribute-match induction* is the task of identifying pairs $\{\langle a_i, a_j\rangle \mid a_i \in A_{\mathcal{E}_1}, a_j \in A_{\mathcal{E}_2}\}$ similar attributes according to a similarity measure, and use those pairs to partition the attribute name space $(A_{\mathcal{E}_1} \times A_{\mathcal{E}_2})$ in non-overlapping clusters.

In **loosely schema-aware blocking** a schema-agnostic blocking is applied in conjunction with a attribute-match induction approach and adapted to exploit loose schema information. This allows to disambiguate blocking keys according to the attribute group from which they are derived (e.g. tokens "Abram" in Figure 1.1b).

The partitioning of the attribute name space is based on four components: (i) the *value transformation function* (ii) the *attribute representation model*, (iii) the *similarity measure* to match attributes, and (iv) the *clustering algorithm*.

• **The value transformation function**. Performing attribute-match induction on two entity collections $\mathcal{E}_1$ and $\mathcal{E}_2$, each attribute is treated as a tuple $\big\langle a_j, \tau(V_{a_j})\big\rangle$, where $a_j \in A_{\mathcal{E}_i}$ is an attribute name, and τ is a *value transformation function* returning the set of *terms* t_n derived from the values V_{a_j} that an attribute a_j can assume in $\mathcal{E}_i$. The function τ generally is a concatenation of text transformation functions *value transformation functions* (e.g. *tokenization*, *stop-words* removal, *lemmatization*).

• **The attribute representation model**. After the transformation, the attributes are represented in a Cartesian space, where each dimension correspond to an element of $T_A = T_{a_{\mathcal{E}_1}} \bigcap T_{a_{\mathcal{E}_2}}$, where $T_{a_\mathcal{E}} = \bigcup_{a_i \in A_\mathcal{E}} \tau(V_{a_i})$. Thus, each attribute name a_j is represented as a feature vector $\mathcal{T}_j$ of length $|T_A|$, where the n-th element is the weight of the element $t_n \in T_A$ in the attribute. Common weight employed to

this purpose are [Pap+13] $TF\text{-}IDF(t_n)$ or the *binary-presence* of the element t_n in $\mathcal{T}_i$ (i.e., $weight = 1$ if $t_n \in T_{a_i}$, 0 otherwise).

• **The similarity measure**. Then, each possible pair of attributes $(a_j, a_k) \in (A_{\mathcal{E}_1} \times A_{\mathcal{E}_2})$ is compared according to a *similarity measure* (e.g. Dice, Jaccard, Cosine). Noteice that the similarity measure must be compatible with the attribute model representation; for instance, the *Jaccard* similarity cannot be employed with the $TF\text{-}IDF$ weighting model.

• **The clustering algorithm**. Finally, the result of the comparison is given as input to the *clustering algorithm* that perform the non-overlapping partitioning of the attribute names.

3.1.2 Meta-blocking

The goal of meta-blocking [Pap+14] is to restructure a collection of blocks, generated by redundant blocking technique, relying on the intuition that the more blocks two profiles share, the more likely they match.

Definition 3.2. META-BLOCKING. Given a blocking collection $\mathcal{B}$, *meta-blocking* is the task of restructuring the set of blocks, producing a new blocking collection $\mathcal{B}'$ s.t. $PQ(\mathcal{B}') \gg PQ(\mathcal{B})$ and $PC(\mathcal{B}') \simeq PC(\mathcal{B})$.

In graph-based meta-blocking, a block collection $\mathcal{B}$ is represented by a weighted graph $\mathcal{G}_\mathcal{B}\{V_\mathcal{B}, E_\mathcal{B}, \mathcal{W}_\mathcal{B}\}$ called *blocking graph*. V is the set of nodes representing all $p_i \in \mathcal{E}$. An edge between two entity profiles exists if they compares in at least one block together: $E = \{e_{ij} : \exists p_i, p_j \in \mathcal{E} \mid |\mathcal{B}_{ij}| > 0\}$ is the set of edges; $\mathcal{B}_{ij} = \mathcal{B}_i \cap \mathcal{B}_j$, where $\mathcal{B}_i$ and $\mathcal{B}_j$ are the set of blocks containing p_i and p_j respectively. $\mathcal{W}_\mathcal{B}$ is the set of weights associated to the edges. The weights capture the likelihood of a match; this is at the base of the edge pruning strategies employed to retain only more promising comparisons. At the end of the pruning, each pair of nodes connected by an edge forms a new block. Hence, meta-blocking inherently prevent from redundant comparisons, since two profiles can appear together in the final blocking collection at most once.

Two classes of pruning criteria can be employed in meta-blocking: the *cardinality-based*, which aims to retain the *top-k* edges, allowing an a-priori deter-

mination of the number of comparisons (the *aggregate cardinality*) and, therefore, of the execution time, at the expense of the recall; and the *weighted-based*, which aims to retain the "most promising" edges though a weight threshold.

Both the pruning criteria can be applied either *locally* or *globally*. In the first case, the *top-k* edges and the weight threshold ϑ are computed and applied in a *node-centric* manner, i.e., for each node and its adjacent edges; while in the second case, the *top-k* edges are selected among the whole set of edges, and the threshold ϑ_i is unique for all the edges.

The combination of those characteristics leads to four *pruning schema*:

- *Weight Edge Pruning* (*WEP*) discards all the edges lower than ϑ.
- *Cardinality Edge Pruning* (*CEP*) sorts all the edges by their weights in descending order, and retains only the first *K*.
- *Weight Node Pruning* (*WNP*) considers singularly each node n_i and its adjacent edges, and prunes those edges that are lower than a local threshold ϑ_i.
- *Cardinality Node Pruning* (*CNP*) similarly to WNP is node centric, but instead of a weight threshold it employs a cardinality threshold k_i (i.e., retain the top k_i edges for each node).

The *weighted-based* pruning criteria have been proved to outperform the *cardinality-based* in terms of PC, but at the expense PQ [Pap+14]. Moreover, existing cardinality based approaches employ an heuristic cardinality threshold selection process, which cannot ensure optimal results for the underlying graph weighting methods. In fact, the threshold for CEP is equals to half the sum of the cardinalities of the blocks in the blocking collection; but that means that the threshold is linearly proportional to the number of entity profiles indexed in the blocking collections. So, for instance, adding n entity profiles to the entity collection that do not match with any other entity, the cardinality threshold selected from CEP would increase. Differently, CNP select a fixed number of edges to retain for each node. But, again, if a set of n non-matching entity profiles are added to the collection, for each of them k edges would be retained, regardless to the strength (weight) of the connection with other nodes.

For these reasons, the meta-blocking approach in BLAST (see section 5) focuses on enhancing PQ adopting a WNP criterion.

3.2 Vocabulary-based Topic Detection

I consider an entity based data model. I assume the existence of an atomic domain $\mathcal{A}$. Of course, there may be more than one atomic domain like **String**, **Integer**, **Date**, etc., but for simplicity I consider here only one. I also assume the existence of an infinite set $\mathcal{C}$ of class names, and an infinite set $\mathcal{N}$ of property names.

A *property* is a pair $\langle p, d\rangle$, where $p{\in}\mathcal{N}$ and $d{\in}\{\mathcal{A}\}{\cup}\mathcal{C}$. The part p is referred to as the name of the property and the part d as the *domain*. I denote as $\mathcal{P}$ the set of all possible properties. A *class* is a pair $\langle c, P\rangle$, where $c{\in}\mathcal{C}$, and $P{\subset}\mathcal{P}$ and is finite.

A data source schema is a finite set of classes C, such that, for every $\langle c, P\rangle$,$\langle c', P'\rangle{\in}C$, with $\langle c, P\rangle{\neq}\langle c', P'\rangle{\in}C$, $c{\neq}c'$. In short, it means that in a data source there cannot be two classes with the same class name. For this reason I can consider a class and its class name equivalently and hereafter write c for referring to the class. Furthermore, if $\langle p, d\rangle{\in}P$, either $d{=}\mathcal{A}$ or $d{\in}C$, which means that a property can have an atomic domain or one of the classes of the data source schema.

My model is generic enough to model the popular relational and RDF schemas. A relational database can be modelled by creating a class for every relational table, in which the class name is the name of the table and the set of properties names consists of one property for each table attribute. The name and the domain of each property is the name and domain of the respective table attribute.

The schema of an RDF database can be modelled in a similar way. A class is created for every RDF class. The name of the class is the name of the RDF class, and the set of properties contain one property for every RDF property that has as a subject the specific RDF class. The name of the property is the predicate of the respective RDF property while the domain is the object of the RDF property.

To be able to understand the contents of a data source, I introduce the notion

of a *signature* which is a compact representation of its contents. Using the schema directly as a signature is not the best choice because schemas are prone to heterogeneity issues. It is common the situation in which the same term has been used for expressing two different semantics, or the situation in which the same semantics have been modelled through different terms. A signature should go beyond the name choices made by the data source designer and be more robust to name variations. An important feature that a signature should capture is the structure. The way data is structured in a data source is not random. It is the way the data administrator decided that the semantics of the data are best expressed. For instance, the reason that two properties are found in the same class is most likely because they model two different aspects of the same real world concept that the class models, and they are both needed for better describing that real world concept. This means that the signature should also capture not only what properties appear in every class but also the co-appearance of the properties. These two principles drive the definition of the signature for classes.

Definition 3.3. SIGNATURE OF A CLASS. The signature of a class $\langle c, P\rangle$ is a graph $G(V, E, f)$, such that its set of nodes being V=$\{c\}\cup N^P$, with N^P=$\{x \mid \exists\langle x, d\rangle\in P\}$, and its set of edges being E=$E^{CP}\cup E^{PP}$, with E^{CP}=$\{\langle c, n_p\rangle \mid n_p\in N^P\}$, E^{PP}=$\{\langle n_p, n'_p\rangle \mid n_p,n'_p\in N^P\}$, and with a function f being an identifier function $f|N^P\cup E^{PP}\rightarrow\mathbb{R}$.

Intuitively, a class signature is a graph that contains one node representing the class, referred to as the *class node*, and one node for every property that the class has, referred to as the *property node*. The graph has an edge between the class node and every property node, referred to as *CP* edges, standing for *Class-Property* edges. It also has one edge between every pair of property nodes, referred to as the *PP* edges, standing for *Property-Property* edges. Finally, every property node in N^P and every *PP* edge in E^{PP} is annotated with numeric value returned by the f function for that edge.

The numeric value plays a role of an identifier for a property and an identifier of the association that exists between two properties of the same class.

In what follows, for simplicity, instead of using the notation $G(V, E, f)$ for a class signature, I will use instead the equivalent more analytic form

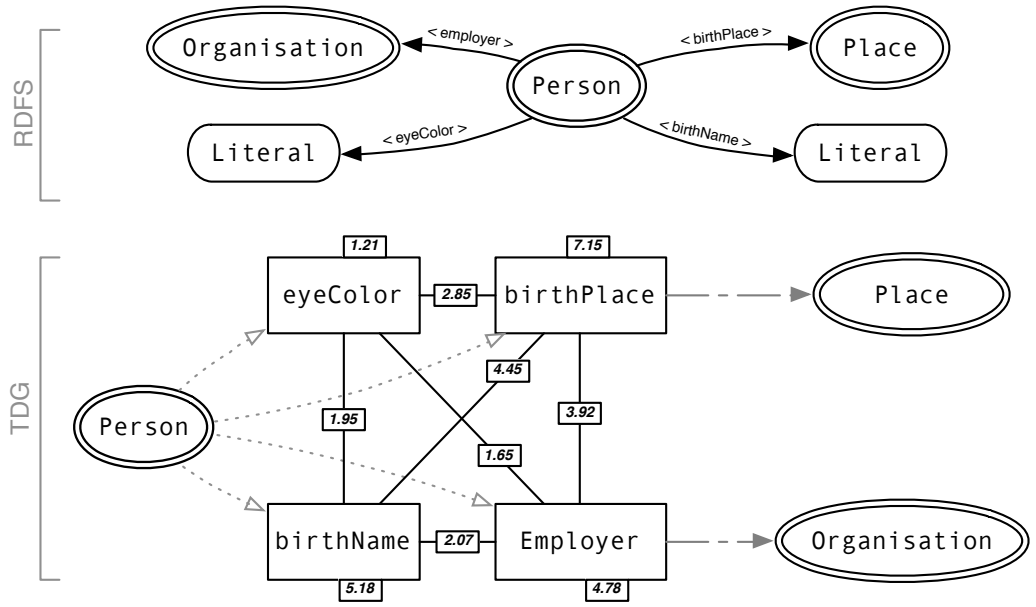


Figure 3.1: A simple RDF schema and its signature.

$\langle c, N^P, E^{PP}, E^{PN}, f\rangle$.

A data source is a set of classes but these classes are not completely independent of one another. They may describe complementary information and there are mechanisms to connect them. In the relational model for instance, such mechanisms are the foreign key constraints. A similar mechanism exists also in RDF. In particular, the value of an RDF property may be not an atomic value, but a URI referencing another RDF entity. In my model this is achieved by properties that have as a domain another class. Thus, to accommodate this important information I consider in the signature of a data source, apart from the class signatures, a set of edges that associate a property node of one class with another class.

Definition 3.4. A *data source signature* is a graph $\langle C, E^{PC}\rangle$ where C is a set of class signatures, and E^{PC} is a set of edges of the form (s, e) such that $s{\in}N^P$ for a $\langle c, N^P, E^{PP}, E^{PN}, f\rangle{\in}C$, and also $e{=}c'$ for a $\langle c', N^{P'}, E^{PP'}, E^{PN'}, f\rangle{\in}C$

Intuitively, a data source signature is a collection of class signatures with an additional set of edges between a property node and a class node. I refer to these edges as PC edges, which stands for *Property-Class edges*, and denote them as E^{PC}.

Example 3.1. Figure 3.1 illustrates a small RDF Schema and its corresponding source signature. The source signature consists of three class signatures (only one illustrated fully). The class nodes are illustrated with double oval lines and the property nodes with squared boxes. The dashed grey lines are the PC edges and the dotted the CP edges. Finally the dotted edges are the PP edges. Note how the E^{PP} and the property nodes are annotated with the identification numbers.

Vocabulary-based Topic Detection. My goal is to understand the topics of a data source. To do so, there is a need for some reference vocabulary in the domain of interest in which the concepts of the data source will be expressed. The reference vocabulary is a collections of classes and possible associations between them. In some sense it can be seen as a data source. It may be provided by a domain expect explicitly or may be a reference data source.

To express the data source in the reference vocabulary, I need to express the source and the vocabulary in some common terminology, in order to match their contents. For this, signatures can be used. Thus, I define the process of identifying the topics of a data source as:

Definition 3.5. VOCABULARY-BASED TOPIC DETECTION. Vocabulary-based topic detection is the process of identifying topics of a data source S with respect to a reference vocabulary S_{ref}.

In my approach vocabulary-based topic detection consists of two sub-tasks: (i) the first to generate the respective signatures $sig_{S_{ref}}$ and sig_S, and (ii) the second to match these two signatures to identify correspondences between their respective components, i.e., pairs of the form $\langle p, t\rangle$ such that $t{\in}S_{ref}$ and $p{\in}c$, with $c{\in}S$.

Chapter 4

Loose Schema Information Extraction

In this chapter, the techniques employed to extract loose schema information- are presented. Firstly (section 4.1), I present attribute-match induction techniques, which can be employed to enhance classical schema-agnostic blocking techniques; secondly (section 4.2), I introduce the technique to build the signatures of the datasource in vocabulary-based topic detection.

4.1 Attribute-match Induction

In the approach proposed in this thesis, to support blocking I propose an entropy extraction criterion applied in combination to an attribute-match induction technique that can be either the *Attribute Clustering* [Pap+13] or the *Loose Attribute Matching*, presented here, and the optional LSH-based support step.

4.1.1 Loose Attribute-matching Induction

Following the definitions of the section 3.1.1, Loose attribute-Match Induction (LMI) is composed of these four components: the *tokenization* as value transformation function; the *binary-presence* of a token as weight for the attribute representation model; the *Jaccard* coefficient as similarity measure; and the algorithm

1 for clustering.

Basically, the algorithm 1 first collects the similarities of all possible attribute pairs of two entity collections, and their maximum values of similarity (lines 2-8). The $similarity$ function (line 4) measure the Jaccard coefficient, i.e., the ratio between the cardinalities of the intersection and the union of two sets: $\frac{|\mathcal{T}_1 \cap \mathcal{T}_2|}{|\mathcal{T}_1 \cup \mathcal{T}_2|}$. Then, (lines 9-13) LAM marks as *candidate* match of an attribute each attribute that is "nearly similar" to its most similar attribute by means of a threshold (e.g.: $0.9 \cdot maxSimilarityValue$). If an attribute a_i has attribute a_j among its candidates, then the edge $\langle a_i, a_j \rangle$ is collected (lines 14-16). Finally, the connected components of the graph built with these edges, with cardinality greater than one, represent the clusters (line 17). Optionally, a *glue*-cluster can gather all the singleton component, as in [Pap+13], to ensure the inclusion of all the possible tokens (blocking keys).

It is important to notice that LMI, as AC [Pap+13], is substantially different from traditional schema-matching approach. In fact, the goal of LMI is not to produce exact matches among attributes of two different datasets, detecting equivalence, hierarchies, and containments of the attributes; but rather to obtain an approximate knowledge of the schemas. Moreover, in attribute-match induction techniques the schema semantic is never involved.

4.1.2 LSH-based Attribute-Match Induction

The computation of the similarity of all possible pairs of attributes has an overall time complexity of $\mathcal{O}(N_1 \cdot N_2)$, where N_1 and N_2 are the cardinality of $A_{\mathcal{E}_1}$ and $A_{\mathcal{E}_2}$ respectively. For the dimensions commonly involved in the semi-structured data of the Web (the data sources schema can commonly have even thousands of attributes) this is an unbearable process. However, only a few (or none) similar attribute profiles are expected to be found similar for each attribute; therefore, employing techniques able to group the attribute approximatively on the basis of their similarity can significantly reduce the complexity of the attribute-match inductions, without affecting the quality of the results. Hence, in BLAST I introduce a pre-processing step that can be optionally employed with both LAM and Attribute Clustering (AC).

Input: Attributes names: A_1, A_2; Attributes values: $\mathcal{T}_1, \ldots, \mathcal{T}_z$
Output: Set of attribute names clusters: K

$edges \leftarrow \{\}$
$Sim \leftarrow Map\langle K, V\rangle$
$Max \leftarrow Map\langle K, V\rangle$
$Cand \leftarrow Map\langle K, \{V\}\rangle$
// most similar attribute for each attribute
foreach $a_i \in A_1, a_j \in A_2$ **do**
    $Sim \leftarrow (\langle a_i, a_j\rangle, similarity(\mathcal{T}_i, \mathcal{T}_j))$
    if $Sim.get(\langle a_i, a_j\rangle) > Max.get(a_i)$ **then**
        $Max \leftarrow (a_i, sim)$
    end
    if $Sim.get(\langle a_i, a_j\rangle) > Max.get(a_j)$ **then**
        $Max \leftarrow (a_j, sim)$
    end
end
// matching attribute candidates generation
foreach $a_i \in A_1, a_j \in A_2$ **do**
    if $Sim.get(\langle a_i, a_j\rangle) \geqslant (\alpha \cdot Max.get(a_i))$ **then**
        $Candidates \leftarrow (a_i, a_j)$
    end
    if $Sim.get(\langle a_i, a_j\rangle) \geqslant (\alpha \cdot Max.get(a_j))$ **then**
        $Candidates \leftarrow (a_j, a_i)$
    end
end
foreach $a_i \in A_1, a_j \in Candidates.get(a_i)$ **do**
    if $a_i \in Candidates.get(a_j)$ **then**
        $edges \leftarrow \langle a_i, a_j\rangle$
    end
end
$K \leftarrow getConnectedComponentsGreaterThan1(edges)$
return K

Algorithm 1: Loose Attribute Matching

LSH (*Locality-Sensitive Hashing* [Bro97]) allows to reduce the dimensionality of an high-dimensional space, preserving the similarity distances, reducing significantly the number of the attribute profile comparisons. Employing the attribute representation model of LMI[1] and Jaccard similarity, *MinHashing* and *banding* [LRU14] can be adopted to avoid the quadratic complexity of comparing all possible attribute pairs.

The set of attributes is represented as a matrix, where each column is the feature vector $\mathcal{T}_j$ of the attirbute a_j (see section 3.1.1). Permuting the rows of that matrix, the *minhash* value of one column is the element of that column that appears first in the permuted order. So, applying a set of n hashing function to permute the rows, each column is represented as vector of n minhash; this vector is called *minhash signature*. The probability of yielding the same minhash value for two column, permuting their rows, is equal to the Jaccard similarity of them; thus, MinHashing preserves the similarity transforming the matrix, with the advantage of having reduced the dimension of the vectors representing the attributes. However, even for relatively small n, computing the similarity of all possible minhash signature pairs may be computationally expensive; therefore, the signatures are divided into *bands*, and only those signatures that are identical in at least one band are considered to be *candidate pairs* and given as input to the attribute-match induction algorithm (adapted to iterate only thorough these candidate pairs - instead of all possible pairs).

Considering n minhash values as signature, b bands for the *banding* indexing, and $r = n/b$ rows for band, the probability of two attributes to identical in at least one band is $1-(1-s^r)^b$. This function has a characteristic *S*-curve form (example Figure 4.1), and its inflection point represents the threshold of the similarity. The threshold can be approximated to $(1/b)^{1/r}$. For instance, choosing $b = 30$ and $r = 5$, the attribute pairs that have a Jaccard similarity greater than ~ 0.5 are considered for the attribute-match induction, otherwise no.

[1]The LMI attribute representation model can be used with *Attribute Clustering* [Pap+13] as well.

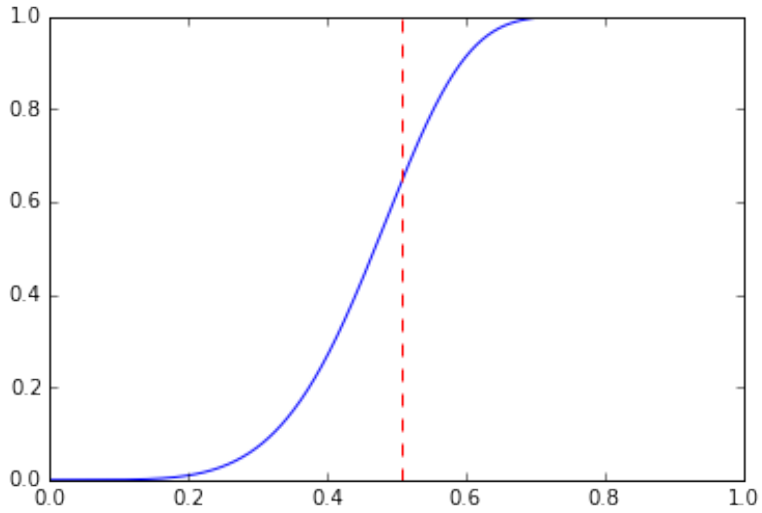


Figure 4.1: LSH S-curve for $r = 5$ and $b = 30$, the dashed line represents the estimated threshold.

4.1.3 Entropy Extraction for Cluster of Attributes

To characterize each attribute cluster generated during the attribute-match induction, BLAST employs the Shannon *entropy* of its attributes.

The entropy of an attribute is defined as follows [CT12]:

Definition 4.1. ATTRIBUTE ENTROPY. Let X be an attribute with an alphabet $\mathfrak{X}$ and consider some probability distribution $p(x)$ of X. I define the attribute entropy $H(X)$ by:

$$H(X) = -\sum_{x \in \mathfrak{X}} p(x) \log p(x)$$

Intuitively, entropy represents a measure of *information content*: the higher the entropy of an attribute, the more significant is the observation of a particular value for that attribute. In other words, if the attribute assumes *predictable* values (e.g., there are only 2 equiprobable values), the observation of the same value in two different entity profiles does not have a great relevance; on the contrary, if the attribute has more *unpredictable* values (e.g., the possible equiprobable values are 100), observing two entity profiles that have the same value for that attribute can be considered a more significant clue for entity resolution.

In BLAST the importance of a blocking key is proportional to the entropy of the attribute from which it is derived. This is obtained weighting the blocking graph according to the entropies (shown in section 5.3). To do so, an entropy value for each group of attribute is derived computing aggregate entropy. The *aggregate entropy* of a group of attributes C_k is:

$$\bar{H}(C_k) = \frac{1}{|C_k|} \cdot \sum_{A_j \in C_k} H(A_j)$$

When a schema-agnostic blocking (e.g. Token Blocking) is applied in combination with attribute-match induction, each blocking key b_i is univocally associated with a cluster C_k, $b_i \mapsto C_k$. For instance, considering the example of Figure 1.1, the token "*Abram*", disambiguated with attribute-match induction, can represent either the blocking key "*Abram_c1*" associated with the cluster C_1, or the blocking key "*Abram_c2*" associated with the cluster C_2; where C_1 is composed of the attributes *Name* of p_1 and *FullName* of p_3, while C_2 is composed of the attributes *addr.* of p_2 and *Address* of p_4.

For meta-blocking, BLAST employs $h(\mathcal{B}_j)$ the entropy associated with a set of blocking key $\mathcal{B}_j$:

$$h(\mathcal{B}_j) = \frac{1}{|\mathcal{B}_j|} \cdot \sum_{b_i \in \mathcal{B}_j} h(b_i)$$

where $h(b_i) = \bar{H}(C_k)$ is the entropy associated to a blocking key $b_i \mapsto C_k$

4.2 Likelihood Estimation of Schema Signatures

Entropy can be employed to measure how informative an attribute (or a cluster of attributes) is, but can be also seen as a characteristic value that can be "almost" identify the attribute from which its derived. This is the idea lying behind the approach to extract the signatures of data sources presented in this thesis.

In fact, a characterization of signatures is carried out by determining a weighted graph that summarizes the subjects addressed by a source. In this thesis I propose to assign the weights in such graphs by using two basic information-theoretical quantities: entropy and mutual information [KN03].

Definition 4.2. Let X be a random variable representing an attribute with alphabet $\mathcal{X}$ and probability mass function $p(x|\theta)$, $\theta \in \Theta \subseteq \mathbb{R}^p$. The entropy $H(X)$ is defined by

$$H(X) = -E_X \log p(X|\theta) \tag{4.1}$$

where $E(\cdot)$ denotes expectation with respect to $p(x|\theta)$.

Note that the above definition does not involve realized values for data instances, thus making the signature independent of the class represented. In particular, entropy describes the uncertainty of values in an attribute. Thus, one problem is estimation of $H(X)$ from available data instances by means of some appropriate approximation of $p(x|\theta)$. If n instances of X are available, then an estimate $\hat{\theta}$ can be obtained by some statistical estimation method, such as maximum likelihood estimation, so that $H(X)$ could be estimated by using $\theta = \hat{\theta}$ in the definition above. To measure the information shared by two attributes at the time I introduce the concept of mutual information.

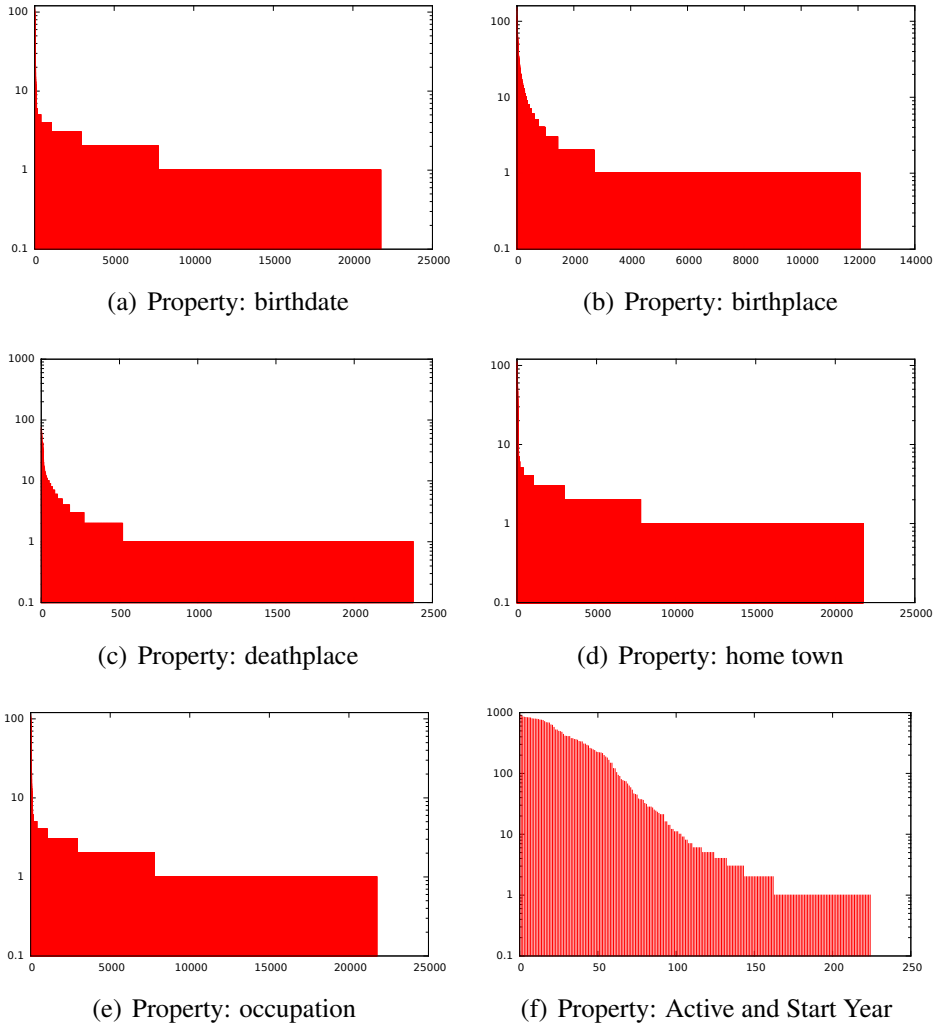
Definition 4.3. Let X and Y be two random variables representing attributes with alphabets $\mathcal{X}$ and $\mathcal{Y}$ with joint mass function $p(x, y|\theta^{XY})$ and marginal mass functions $p(x|\theta^X)$ and $p(y|\theta^Y)$. The mutual information of X and Y and Y is:

$$\begin{aligned} I(X;Y) &= E_{XY}\left[\log \frac{p(X,Y|\theta^{XY})}{p(X|\theta^X)p(Y|\theta^{XY})}\right] \\ &= H(X) + H(Y) - H(X,Y) \end{aligned} \tag{4.2}$$

where $H(X)$ and $H(Y)$ are entropies for X and Y and $H(X,Y)$ is the entropy for the pair (X,Y).

Note that $I(\cdot;\cdot)$ measures different levels of association (or shared information) between pairs of nodes. Moreover, similarly to entropy, also the mutual information needs to be estimated from data instances. To estimate $I(X;Y)$ I need to obtain parameter estimates $\hat{\theta}^X$, $\hat{\theta}^Y$, and $\hat{\theta}^{XY}$.

The method for estimating $H(X)$, $H(Y)$ and $H(X,Y)$ from the data is crucial to obtain representative signatures. A suitable method should be able to prevent over-fitting. The estimated signature does not have to perfectly replicate a specific data source, but rather provide us with a summarized representation of the concepts described in it. Over-fitting is important in the presence of very large al-



(a) Property: birthdate

(b) Property: birthplace

(c) Property: deathplace

(d) Property: home town

(e) Property: occupation

(f) Property: Active and Start Year

Figure 4.2: Frequency distribution of some properties of the DBpedia Musical Artist class.

phabets for the attributes under examination, with only a few observed instances. The elements of the alphabets with very low frequency typically inflate the overall noise thus deteriorating the quality of the available information.

In the analysis of real data sources, I noticed that the contribution of such low frequency elements is not negligible. Let us consider for example Figure 4.2 that shows through Pareto charts the frequency distribution of some properties of the DBpedia Musical Artist class. Each figure shows in the x axis the different elements of the alphabet and in the y axis the respective frequency. It is evi-

dent that frequency distributions are usually right skewed, thus meaning that only few elements of the alphabets really contribute, with their high frequency, in the characterization of the entropy value for the specific property. Even if Figure 4.2 includes only some properties of a selected DBpedia class, I performed an extensive analysis on the other classes and properties, obtaining similar frequency distributions.

Moreover, (4.1) shows that the entropy value does not have a upper bound value, since it depends on the cardinality of the attribute alphabet. As a consequence, attributes having a different number of alphabet elements in different dataset have different entropy values even when they represent the same real world object. In Section 7.2, I will show that this problem has a big impact in real data sources, and it can affect the result accuracy. For this reason, a normalizing factor that limits the range of the entropy and mutual information values is needed.

Finally, another issue in using entropy and mutual information values is related to the high dimensionality of the problem, that has a big impact on the time complexity. The high number of instances usually collected in the databases available online makes the calculation of the actual values expensive. For example, if I adopt the DBpedia Ontology as vocabulary, the class Person (one of the 529 classes which form a subsumption hierarchy) of the DBpedia Ontology contains 832.000 instances and has 101 properties (in version 3.9). This means that the cardinality of the set E^{PP} built considering only the class Person is 5,050.

Chapter 5

Blocking with Loosely Schema-aware Techniques

In this chapter, my novel (meta-)blocking approach is presented. I called it BLAST (Blocking with Loosely-Aware Schema Techniques). Fundamentally, it consists of the holistically combination of loosely schema-aware blocking and meta-blocking for clean-clean Entity Resolution. In particular, BLAST operates in three main phases: (i) the loose schema information extraction, (ii) the blocking, and (iii) the meta-blocking.

5.1 Loosely Schema-aware Information Extraction

The first phase consists in the loose schema information extraction: while an attribute-match induction approach is employed to obtain information about the attribute similarities (as described in section 4.1.1), the entropies of the attribute values are collected (as described in section 4.1.3) in order to be exploited at a later stage for meta-blocking. An LSH pre-processing step (section 4.1.2) optionally supports the attribute-match induction approach; this allows to reduce the complexity the high dimensional space of the *attribute representation model* when dealing with data sources characterized by high number of attributes.

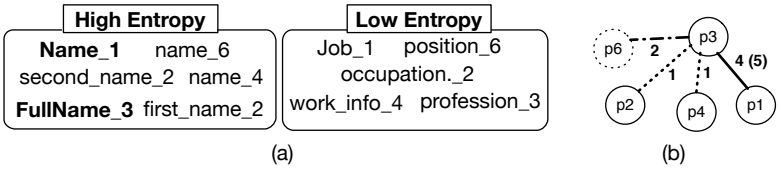


Figure 5.1: High and low entropy group of attributes.

5.2 Loosely Schema-aware Blocking

In the second phase, the induced information about clusters of similar attributes is exploited to enhance the efficiency of a schema-agnostic technique. In particular, in BLAST I employ Token Blocking[1] following [Pap+13]: each token represents as many different blocking keys as many are the clusters that contain an attribute in which it appears, while in classical Token Blocking each token represents a unique blocking key. The example in Figure 1.1b gives an intuition of the benefits of this approach: being able to disambiguate the token "Abram" according to the attribute in which it appears, preventing to index together non-matching profiles, and avoiding 4 superfluous comparisons out of the 6 possible.

5.3 Loosely Schema-aware Meta-blocking

In the third phase, meta-blocking is performed: the blocking graph $\mathcal{G}_\mathcal{B}$ corresponding to the resulting blocking collection is generated, weighted taking into account the co-occurrences of the entity profiles and the entropies of the attributes previously computed. In particular, the χ^2 test is employed to measure the strength of co-occurrences. Then, each edge weight is re-weighted according to the *aggregate entropy* associated to it.

To give an intuition of the benefits yielded by this latest step, consider the example in in Figure 5.1. Without considering the entropies, applying a node-centric pruning (e.g., selecting as weight threshold the average of the local weights) on the p4 node-centric view of the blocking graph (Figure 5.1 b), entails the retention of the non-matching edge (i.e., a superfluous comparison) $p_3 - p_6$. A way to avoid

[1] Other techniques [Pap+15] can be adapted to this scope as well, but comparing them is out of the scope of this thesis.

this, is increase the score of the edges derived from blocking keys associated to attributes with high entropy. For instance, considering the classification listed in Figure 5.1a (listing an high entropy attribute cluster), and assigning bonus equals to +1 to each edge associated to attributes with high entropy.

In the following the details of Loosely Schema-aware meta-blocking are described. In the first step, the unweighted version of the blocking graph is built; then, the weights of the blocking graph $\mathcal{G}_\mathcal{B}\{V_\mathcal{B}, E_\mathcal{B}, \mathcal{W}_\mathcal{B}\}$ are computed according to a *weighting schema*. The second step consists in the pruning.

5.3.1 Blocking Graph Weighting

Considering two entity profiles p_u and p_v, the contingency table describing their joint frequency distribution in a given blocking collection is shown in Table 5.1. The table describes how entity profiles p_u and p_v co-occur in a blocking collection. For instance: the cell o_{12} represents the number of blocks in which p_u appears without p_v (the absence is denoted with $\neg$); the cell o_{2+} represents the number of blocks in which p_u is not present (independently of p_v). These values are also called *observed* values. As an example, the values in parentheses are values derived form the blocking collection of figure 1.1b for the profiles p_1 and p_3.

Given this representation, BLAST employs Pearson's chi-squared test (χ^2) [AK11] to quantify the independence of p_u and p_v in blocks; i.e., testing to see if the distribution of p_v given that p_u is present the blocks (first row of the table) is the same as the distribution of p_v given that p_u is not present (the second row in the table). In practice, the chi-squared test measures the divergence of observed (o_{ij}) and expected (e_{ij}) sample counts (for $i = 1, 2, j = i, 2$). The expected values are with reference to the null hypothesis, i.e., assuming that p_u and p_v appear independently in the blocks. Thus, the expected value for each cell of the contingency table is: $e_{ij} = \frac{o_{i+} \cdot o_{+j}}{n_{++}}$.

Hence, the weight w_{uv} associated to the edge between the nodes representing

	p_v	$\neg p_v$	
p_u	o_{11}	o_{12}	o_{1+}
$\neg p_u$	o_{21}	o_{22}	o_{2+}
	o_{+1}	o_{+2}	o_{++}

Table 5.1: Contingency table for p_u, p_v.

	p_3	$\neg p_3$	
p_1	4	1	5
$\neg p_3$	1	2	3
	5	3	8

Table 5.2: Contingency table for p_1 and P_3 from figure 1.1b.

the entity profiles p_u and p_v is computed as follow:

$$\begin{aligned} w_{uv} &= \chi^2_{uv} \cdot h(\mathcal{B}_{uv}) \\ &= \sum_{i \in \{1,2\}} \sum_{j \in \{1,2\}} \frac{o_{ij} - e_{ij}}{e_{ij}} \cdot h(\mathcal{B}_{uv}) \end{aligned} \tag{5.1}$$

Notice that BLAST is using the test statistic as a measure that helps to highlight particular profile pairs (p_u, p_v) that are highly associated in the blocking collection, and not to accept or refuse a null hypothesis. The correcting entropy value just weight the importance of the blocks in which a co-occurrence appears, since not all the blocks are equally important (as discussed in section 4.1.3).

5.3.2 Graph Pruning

Selecting the pruning threshold is a critical task. To achieve the highest level possible of PQ, and perform fine-grained pruning on the blocking graph, BLAST adopts a WNP schema (see section 3.1.2). I identify a fundamental characteristic that threshold selection method, in WNP, must present: the independency of the local number of adjacent edges, to avoid the sensitivity to the number of *low-weighted* edges in the blocking graph. In fact, this issue arises when employing threshold selection functions that depend on the number of edges, such as the *average* of the weights [Pap+14].

Example 5.1. To illustrate this phenomenon, consider again the example in Figure 1.1. Figure 1.1c depicts the blocking graph $\mathcal{G}_\mathcal{B}$ generated from the blocking collection of Figure 1.1b, and Figure 1.1e shows a subgraph $\mathcal{G}_{p_1}$, which represents the *node-centric* view of the $\mathcal{G}_\mathcal{B}$ for the entity profile p_1 (Figure 1.1c). If the entity collection (Figure 1.1a) is composed only of the profile set (p_1, p_2, p_3, p_4), the resulting graph $\mathcal{G}_{p_1}$ has only 4 nodes and 4 edges. In this scenario the average of the edge weights (the threshold) is slightly greater than 2 and only the edge between p_1 and p_3 is retained. But, if the entity profile p_5 is added to the entity collection, one node and one edge are added to $\mathcal{G}_{p_1}$, influencing the threshold that became 2 and implying the retention of a second edge (between p_1 and p_4). Therefore, the comparison of p_1 and p_4 depends on the presence or absence of p_5 in the entity collection, even though the similarity between those two profiles does not depends on p_5.

Hence, in BLAST I introduce two *weight threshold selection schemes* independent of the number of edges in the blocking graph:

Average Min Max (AM2). AM2 is the mean between the local upper bound $m_i \equiv min(W_i)$ and lower bound $M_i \equiv max(W_i)$, where W_i is the multiset of the adjacent edge weights; unlike WM, it needs no parameter selection:

$$\begin{aligned} \vartheta_i &= \frac{m_i + M_i}{2} \\ &= m_i + \frac{M_i - m_i}{2} \end{aligned} \tag{5.2}$$

Average Min Max Maximized (AM3). In AM3 weights must be normalized with respect to the maximum weight of the underlying schema; then, the average computed as fo AM2 is weighted in accord to the "strength" of the upper bound. Intuitively, if the upper bound is close to 1, it means that the underlying blocking technique and weighting function are capturing with a low uncertainty the true matching pairs of profiles, hence the threshold can be increased; otherwise, if the upper bound is low, the uncertainty is greater and a more conservative threshold is more appropriate:

$$\vartheta_i = m_i + \frac{(M_i - m_i) \cdot \alpha}{2}; \qquad \alpha = 1 + M_i \tag{5.3}$$

Chapter 6

Vocabulary-based Topic Detection with Loose Schema Information

In this chapter I present a novel approach to identify the topics of a data source given a reference vocabulary, exploiting loosely schema-aware techniques. All the process is based on the building of distinctive signatures for both the target data source(s) and the reference vocabulary. I adopt the signature representation model introduced in section 3.2. Section 6.1 describes how entropy and mutual information (and their variants) are extracted to be employed as node and edge weight respectively in the signatures. Then, in section 6.2, the signature matching process is presented. Finally, section 6.3 introduces WHATSIT, the implementation of my approach.

6.1 Computing Entropy and Mutual Information for Data Source Signature

To weight a data source signature, I implemented and tested three measures based on entropy and the mutual information: (i) a classical implementation based on Shannon's logarithmic entropy that provides us with a benchmark for the comparisons; (ii) a weighted entropy and mutual information implementation, that allows us to remove some "noise" provided by large regions of values with low probabilities and (iii) the pseudo-additive entropy and mutual information developed

by Havrda and Charvát [HC67]. [Tsa88] has exploited the stability of pseudo-additive in the context of statistical mechanics. Differently from previous work in this area, I considered appropriate normalizations for all the above entropy measures. As a result the range for all the measure is between 0 and 1, which allows a more fair comparison of their performance.

I begin by estimating parameters for the attributes from N available instances. In this thesis I assume a multinomial distribution for the attributes, but my approach can be easily extended to other statistical models. Suppose that a single attribute X_i, $i = 1, \ldots, k_i$ follows the multinomial model $X_i \sim \text{Mult}(\theta_i)$, where θ_i is the k_i-vector $\theta_i = (\theta_{i1}, \ldots, \theta_{ik_i})^T$, and θ_{ij} corresponds to the probability of the j^{th} element of attribute alphabet, and $\sum_{j=1}^{k_i} \theta_{ij} = 1$. For instance, in RDF data sources, the alphabet of an attribute is the set of URIs and literals associated through the `rdf:range` statement of a property.

Then, given N observations on X I have the following likelihood function

$$p(x_{i1}, \ldots, x_{ik_i}|\theta) = \frac{N!}{x_{i1}! \cdots x_{ik_i}!} \theta_{i1}^{x_{i1}} \cdots \theta_{ik_i}^{x_{ik_i}}, \tag{6.1}$$

where x_{ij} denotes the number of times I observe the jth element of the alphabet and $\sum_{j=1}^{i_k} x_{ij} = N$. From (6.1), the maximum likelihood estimates for θ_{ij} is simply the frequency $\hat{\theta}_{ij} = x_{ij}/N$, for all $j = 1, \ldots, k_i$.

Using estimated parameters, I estimate the marginal entropy for X_i for all $i = 1, \ldots, k_i$ based on the following measures:

i) $\hat{H}_i^S = -\sum_{j=1}^{k_i} \hat{\theta}_{ij} \log \hat{\theta}_{ij}, \;\; i = 1, \ldots, k_i;$ (Shannon Entropy) (6.2)

ii) $\hat{H}_i^W = -\sum_{j=1}^{k_i} \frac{\hat{\theta}_{ij}^a \log \hat{\theta}_{ij}}{\sum_{j=1}^{k_i} \hat{\theta}_{ij}^a}, \;\; a > 0, \; i = 1, \ldots, k_i;$ (Re-weighted Entropy) (6.3)

iii) $\hat{H}_i^P = -\sum_{j=1}^{k_i} \hat{\theta}_{ij} \frac{1 - \hat{\theta}_{ij}^b}{b}, \;\; b > 0, \;\; i = 1, \ldots, k_i.$ (Pseudo-additive Entropy) (6.4)

The weighted entropy (6.3) is based on a simple variation of the classical measure, where the components are "weighted" by means of a parameter a. For the pseudo-additive entropy measure, convexity is achieved only for $b > 0$. These are empirical parameters defined, on the basis of the experiments. In the evaluated data sets, the best results were achieved using values for a and b near 1.5 and 0.5, respectively. Note that the form in $\hat{H}_i^W$ and $\hat{H}_i^P$ reduce the noise generated by elements with low frequency and to improve the contribution by the high frequency elements.

A normalization factor for the above measures is obtained by computing the maximum entropy. This can be achieved by replacing $\hat{\theta}_{i1}, \ldots, \hat{\theta}_{ik_i}$ with the uniform distribution $1/k_i, \ldots, 1/k_i$. Straightforward algebra gives the maximum values for i), ii) and iii); respectively, $\hat{H}_{i,max}^S = \log k_i$, $\hat{H}_{i,max}^W = \log k_i$ and $\hat{H}_{i,max}^P = (1 - k_i^{-b})/b$. Finally, to normalize I simply divide each entropy measure by its maximum value.

To evaluate the relationship between two attributes, say X_i and X_l, I use the following measures of mutual information which are derived from the entropy measures above. Specifically,

$$\text{i)}\ \hat{MI}_{il}^S = \sum_{j=1}^{k_i}\sum_{l=1}^{k_l} \hat{\theta}_{ijlm} \log\left(\frac{\hat{\theta}_{ijlm}}{\hat{\theta}_{ij}\hat{\theta}_{lm}}\right),\ \ 1 \le i \le k_i, 1 \le l \le k_l; \tag{6.5}$$

$$\text{ii)}\ \ \hat{MI}_i^W = \sum_{j=1}^{k_i}\sum_{l=1}^{k_l} \frac{\hat{\theta}_{ijlm}^a}{\sum_{j=1}^{k_i}\hat{\theta}_{ijlm}} \log\left(\frac{\hat{\theta}_{ijlm}}{\hat{\theta}_{ij}\hat{\theta}_{lm}}\right),\ \ a > 0,\ \ 1 \le i \le k_i, 1 \le l \le k_l; \tag{6.6}$$

$$\text{iii)}\ \hat{MI}_i^P = -\sum_{j=1}^{k_i}\sum_{l=1}^{k_l} \hat{\theta}_{ijlm}\left[\frac{1 - \hat{\theta}_{ijlm}/(\hat{\theta}_{ij}\hat{\theta}_{lm})^b}{b}\right],\ \ b > 0,\ \ 1 \le i \le k_i, 1 \le l \le k_l. \tag{6.7}$$

where $\hat{\theta}_{ij}$,$i = 1, \ldots, k_i$, and $\hat{\theta}_{lm} = 1, \ldots, k_l$ denote marginal empirical frequencies for the values of the attributes X_i and X_l respectively, while $\hat{\theta}_{ijlm}$ represents joint empirical frequencies for the values of the attributes X_i and X_l.

6.1.1 Confidence Intervals

In this section I consider the problem of comparing entropies computed from different samples. Matching based solely on point measurements is not sufficiently reliable, due to the presence of statistical error. Thus, I propose to compare entropy measures by constructing confidence intervals for the difference of entropy difference. Specifically, let $\widehat{H} = H(\hat{\theta}_1, \ldots, \hat{\theta}_k)$ be an arbitrary entropy method; specifically consider entropies i), ii) or iii) described in the previous section. Further, denote by $\widehat{H}_1$ and $\widehat{H}_2$, entropies on the same attribute computing based on observations from k_1 and k_2 alphabets, respectively; $\widehat{H}_1$ and $\widehat{H}_2$ are estimated using counts in N_1 and N_2 independent samples.

Let $0 < \alpha < 1$ denote a pre-specified confidence level. A $(1-\alpha)\%$ confidence interval for the true entropy difference is

$$\begin{aligned} \mathcal{CI}(\hat{H}_1, \hat{H}_2, \alpha) = & \frac{\hat{H}_1}{H_{1,max}(m_1)} - \frac{\hat{H}_2}{H_{2,max}(N_2)} \\ & \pm z_{1-\alpha/2} \sqrt{\frac{V(\hat{\theta}_1, \ldots, \hat{\theta}_{k_1}, N_1)}{H_{max}(N_1)^2} + \frac{V(\hat{\theta}_1, \ldots, \hat{\theta}_{k_2}, N_2)}{H_{max}(k_2)^2}}, \end{aligned} \tag{6.8}$$

where $H_{max}(k)$ is the maximum entropy obtained by replacing the probabilities $p_1, \ldots, p_k$ with uniform probabilities $1/k, \ldots, 1/k$; so for the Shannon entropy the maximum value is $\log m$ while for the pseudo-additive entropy I have $(1 - k^{-b})/b$. Further, in the above expression, z_q is the q-quantile for the standard normal distribution and $V_i(\cdot)$ represents an expression for an approximation of the variance of $\hat{H}_i$ obtained by the Delta method [Vaa00] . Particularly, for $j = 1, 2$, I have

$$V_j(\theta_1, \ldots, \theta_{k_j}, N_j) = \frac{1}{N_j} (\nabla \hat{H}_j)^T \begin{pmatrix} \theta_1(1-p_1) & -\theta_1\theta_2 & \cdots & -\theta_1\theta_{k_j} \\ -\theta_1\theta_2 & \theta_2(1-\theta_2) & \cdots & -\theta_2\theta_{k_j} \\ \vdots & \vdots & \ddots & \vdots \\ -\theta_1\theta_{m_j} & -\theta_2\theta_{m_j} & \cdots & \theta_j(1-\theta_{k_j}) \end{pmatrix} \nabla \hat{H}_j,$$

where $\nabla \hat{H}_j = (\partial \hat{H}_j / \partial \theta_1, \ldots, \partial \hat{H}_j / \partial \theta_{k_j})^T$ is the gradient vector of partial derivatives of the entropy function. Clearly, the form of such a vector depends on the

definition of the entropy function. For, example Shannon's entropy I have

$$\nabla \hat{H}_j^S = (\log \theta_1 + 1, \ldots, \log \theta_{k_j} + 1)^T$$

while for the pseudo-additive entropy I have

$$\nabla \hat{H}_j^S = (L_a(\theta_1) + \theta_1^a, \ldots, L_a(p_{m_j}) + p_{k_j}^a)^T,$$

where the function $L_a(u) = (u^a - 1)/a, u > 0, \alpha > 0$ is the generalized logarithm.

6.2 Matching Data Sources Signatures

The goal of the matching process is to find the signatures associated to classes of the target source that match into dependency graphs associated to the reference ontology. As described in Section 3.2, the signatures model classes and properties as graphs where nodes and edges are weighted. Entropy (or pseudo-additive entropy) is used for weighting nodes, mutual information (or pseudo-additive mutual information) for the edges. Nevertheless, the effort required for computing the weights is not the same: the complexity of the mutual information computation grows quadratically with the growing of the number of the class properties, while the complexity of the entropy computation grows linearly.

Even if an accurate matching process should take into account nodes and edges, I decided to design a straightforward two-step process, that requires the computation of the mutual information only when needed, thus reducing complexity in the case that the reference ontology has a high number of properties per class (i.e., avoiding to compare a huge number of possible pairs of properties). Firstly, for each property of the target source, a set of *candidate matching properties* belonging to the reference ontology is computed. The computation of matches requires to take into account the pre-computed entropy stored in GE_{index} and the confidence interval values dynamically computed on the basis of maximum entropies and entropy variances[1].

[1]For the reference ontology, the maximum entropies and entropy variances are stored in GE_{index}; while, for the target source, these measures have to be computed at runtime.

Secondly, mutual information is computed only for those target properties that belongs to more groups, to select the best option. This selection and the computation of the final result can provide two different kinds of results: (a)`1SIG matching`, where the prototype matches the target properties into properties belonging to one single signature in the reference ontology; (b)`1+SIG matching`, where the prototype matches the target properties into properties belonging to several signatures. Obviously, `1SIG matching` is the simplest case since it presumes that target source and reference ontology model the real world in the same way. My technique is able to manage both the options.

Summarizing, my solution allows to perform an entropy-based match in first place, and then disambiguate the match with the support of mutual-information as a second step. This can achieve a sub-optimal result (the optimal solution should consider entropy and mutual information contemporarily), but allows to avoid the O(n^2), with n the number of properties of a class, computation of Mutual Information; I demonstrate in the experiment section 7.2 that this is enough to prove the efficacy of my signature-base approach.

Example 6.1. Let us consider a class C_t of a target source, with five properties $(p_1, p_2, p_3, p_4, p_5)$, and a entropy-based match that returns the following candidate matching properties.

- $p_1 : \{Person_{birthYear}, Band_{startYear}\}$
- $p_2 : \{Person_{deathYear}, Band_{name}\}$
- $p_3 : \{Band_{country}\}$
- $p_4 : \{Person_{height}\}$
- $p_5 : \{\emptyset\}$

Person and *Band* are two classes of the reference ontology. In case of `1SIG matching`, the matches of properties p_1 and p_2 have to be disambiguated via the mutual information, and either p_3 or p_4 are leaved unmatched according to the result of the previous process.

The matching process relies on the $entropy(\bar{b})$ function, that takes as input a vector of discrete property values $\bar{b} = (b_1, b_2, \ldots, b_m)$, and returns its entropy value. $entropy(\bar{b})$ allows us to provide a weight to the edges E^{CP} of the Dependency Graph. I can use now this representation to match the signatures of a target data source. For each data source, I build a multiset $\bar{e_i} = \{e_1, e_2, \ldots, e_l, \ldots, e_m\}$, where each element e_l represents the entropy of the l^{th} property of the class c_i, i.e., $entropy(a_l^{c_i})$. I define the *target match property set* Λe_l and *candidate class set* $c(\Lambda e_l)$ of DBpedia for each $\bar{e_i}$

$$\Lambda e_l = \{a_j^{c_k} \mid c_k \in KB : 0 \in \mathcal{CI}(entropy(a_j^{c_k}), e_l, \alpha)\} \tag{6.9}$$

$$c(\Lambda_{e_l}) = \{c_k \mid \exists a^{c_k} \in \Lambda_{e_l}\} \tag{6.10}$$

where e_l the l^{th} element of $\bar{e_i}$, and α is typically equal to 0.05.

The idea I have implemented to finding the best matches is the maximization of the *Coverage* of the matching classes belonging to the reference ontology. I define $coverage(\cdot)$ with respect to a subset of the classes $K' \subseteq K_{ref}$, where K_{ref} is the set of classes in the knowledge base is (in my case DBpedia) as:

$$cover(e_l, K') = \begin{cases} 1, & \text{if } \exists c_k \in K' \mid a_j^{c_k} \in \Lambda e_l \\ 0, & \text{otherwise} \end{cases} \tag{6.11}$$

$$coverage(\bar{e_i}, K') = \sum_{e_l \in \bar{e_i}} cover(e_l, K') \tag{6.12}$$

Whenever a conflict arises on matching classes, i.e., $|c(\Lambda e_l) \cap c(\Lambda e_t)| \geq 2$, I may compute the mutual information between the properties $a_l^{c_k}$ and $a_t^{c_k}$ (corresponding to the properties having entropies matching with e_l and e_t respectively) for all the classes in $c(\Lambda e_l) \cap c(\Lambda e_t)$. In this case, the approach proceeds greedily, trying to perform MI-based matching with properties of the classes in $c(\Lambda e_l)$, and stopping computation in case a positive match is found. The output of the matching

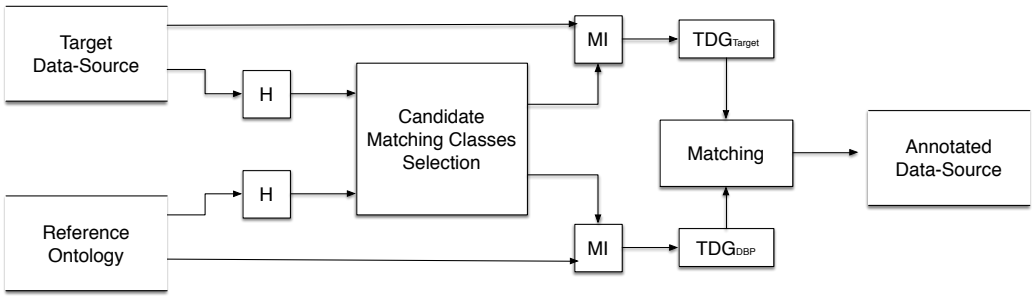


Figure 6.1: The WHATSIT prototype functional architecture.

process is set of class $C^c \subseteq K$, ranked according to the coverage of each class after the MI-based disambiguation phase.

6.3 The WHATSIT prototype

The presented approach has been implemented in a prototype system, called WHATSIT˙WHATSIT has been implemented in `python 2.7`. The functional architecture of WHATSIT is shown in Figure 6.1. WHATSIT takes as an input a populated reference ontology, in my case DBpedia, and a target RDF data source. The output is the target source annotated in each property with the corresponding DBpedia property associated to the domain.

To build and compare the dependency graphs of the sources, WHATSIT has to compute entropies and mutual information of properties. But, the real time computation of mutual information for all pairs of properties is often infeasible. This is because it is common to have RDF data sources with hundreds of properties per class, that would lead to a huge number of pairs. Moreover, computing mutual information for each pair of properties, often is superfluous for the match (see Section 7.2.3). For these reasons, WHATSIT relies firstly on the entropy for identifying candidate matching properties, and computes the mutual information in case a disambiguation is needed. Thus, the matching of the dependency graphs is split in two match phases: the first considering only the entropy, the second considering both entropy and mutual information. Furthermore, entropies of the reference ontology can be computed in a pre-processing step, and stored in a in-

dex that I call PA_{index} (*Pseudo-additive Entropy index)*. To produce candidate matching classes, the match is performed considering a confidence interval for the entropies, that has dynamically computed as described in Section 6.1.

Chapter 7

Experimental Evaluation

7.1 BLAST evaluation

7.1.1 Experimental Setup

Datasets: For the experimental evaluation I employ established benchmarks [Pap+15; Pap+14; PPK14; Pap+13; MT13b; KTR10][1] composed of real-world datasets with different characteristics and volume. I perform six different comparisons, listed in Table 7.1. The number of entities and attribute names in each profile collection is denoted by $|\mathcal{E}|$ and $|A|$ respectively; the number of name-value pairs corresponds to $n - vpairs$; and $|\mathcal{D}uplicates^{\mathcal{E}}|$ represents the total number of actual duplicates. Each comparison consists of a pair of record sets extracted from on-line data sources of different domain (bibliographic, e-commerce, movies, and general): `ar1` matches article profiles from DBLP[2] and ACM[3]; `ar2` matches article profiles from DBLP and Google Scholar[4]; `pr1` matches product profiles from Amazon.com products and Google products[5]; `pr2` matches product profiles from Abt.com and Buy.com; `mv` matches movie profiles from IMDB[6] and

[1] the version employed in [Pap+15]: http://sourceforge.net/projects/erframework/files/ CleanCleanERDatasets/

[2] http://www.dblp.org

[3] http://dl.acm.org

[4] https://scholar.google.com

[5] Extracted with the Google Base Data API

[6] http://www.imdb.com

	fully mappable			
	$\lvert\mathcal{E}_1\rvert$ - $\lvert\mathcal{E}_2\rvert$	$\lvert A_1\rvert$ - $\lvert A_2\rvert$	$n-v$ pairs	$\lvert\mathcal{D}uplicates^{\mathcal{E}}\rvert$
ar1 DBLP ACM	2.6k - 2.3k	4 - 4	10k - 9.2k	2.2k
ar2 DBLP Scholar	2.5k - 61k	4 - 4	10k - 198k	2.3k
pr1 Amazon Google	1.4k - 3.0k	4 - 4	5.3k - 9.1k	1.1k
pr2 Abt Buy	1.1k - 1.1k	4 - 4	2.6k - 2.3k	1.1k

	partially mappable			
	$\lvert\mathcal{E}_1\rvert$ - $\lvert\mathcal{E}_2\rvert$	$\lvert A_1\rvert$ - $\lvert A_2\rvert$	$n-v$ pairs	$\lvert\mathcal{D}uplicates^{\mathcal{E}}\rvert$
mv IMDB DBp	28k - 23k	4 - 7	155k - 816k	23k
dbp DBp07 DBp09	$1.2M$ - $2.2M$	30k - 50k	$17M$ - $35M$	893k

Table 7.1: Datasets characteristics.

DBpedia[7]; dbp matches entity profiles from two different snapshots of DBpedia (2007 and 2009) − it is important to notice that only the 25% of the name-value pairs is shared among the two snapshots, due to the constant changes in DBpedia, therefore the ER is not trivial. The comparison can involve datasets whose attributes can be mapped with either $1{:}1$ associations (i.e., *fully mappable*), or $0{:}n$ associations (i.e., partially mappable).

Evaluation Metrics: I evaluate the quality of the produced blocking collections in terms of precision and recall, through their surrogates PC and PQ (section 3.1). The comparison against a baseline is expressed with

$$\Delta PC(\mathcal{B}, \mathcal{B}') = \frac{PC(\mathcal{B}) - PC(\mathcal{B}')}{PC(\mathcal{B})};$$

$$\Delta PQ(\mathcal{B}, \mathcal{B}') = \frac{PQ(\mathcal{B}) - PQ(\mathcal{B}')}{PQ(\mathcal{B})}$$

where $\mathcal{B}$ is the baseline blocking collection, and $\mathcal{B}'$ is the compared blocking collection. I also consider the F_1 score, defined as:

$$F_1(\mathcal{B}') = 2 \cdot \frac{PC(\mathcal{B}') \cdot PQ(\mathcal{B}')}{PC(\mathcal{B}') + PQ(\mathcal{B}')}$$

[7] http://dbpedia.org

	ar1					ar2				
	PC(%)	PQ(%)	F_1	t_o(s)	t_e(s)	PC(%)	PQ(%)	F_1	t_o(s)	t_e(s)
Unsupervised MB	97.30	7.55	0.139	1.52	0.56	96.29	0.41	0.008	2.85	8.18
LSH-AC+Unsupervised MB	97.33	8.39	0.154	0.88	0.35	96.16	0.41	0.008	4.93	4.41
Supervised MB	92.40	49.42	0.644	5.65	0.10	90.30	11.09	0.198	9.09	0.53
BLAST	96.40	31.68	0.477	0.87	0.11	94.06	1.41	0.028	5.51	1.83

	pr1					pr2				
	PC(%)	PQ(%)	F_1	t_o(s)	t_e(s)	PC(%)	PQ(%)	F_1	t_o(s)	t_e(s)
Unsupervised MB	76.07	1.66	0.033	0.76	5.45	78.14	6.48	0.119	0.19	0.19
LSH-AC+Unsupervised MB	89.91	1.77	0.035	1.85	7.95	78.33	6.45	0.119	0.31	0.21
Supervised MB	67.87	39.06	0.495	4.75	0.09	71.55	22.09	0.338	3.35	0.06
BLAST	79.26	7.76	0.142	1.71	0.58	76.30	20.78	0.327	0.28	0.07

	mv					dbp				
	PC(%)	PQ(%)	F_1	t_o(m)	t_e(m)	PC(%)	PQ(%)	F_1	t_o(h)	t_e(h)
Unsupervised MB	96.10	0.33	0.007	1.05	3.90	97.47	0.06	0.001	16	60
LSH-AC+Unsupervised MB	96.41	0.35	0.007	0.95	3.65	98.67	<0.01	<0.001	21	>100
Supervised MB	92.82	4.24	0.081	3.8	0.83	97.30	0.19	0.004	33	39
BLAST	93.37	14.34	0.249	1.3	0.09	96.27	0.83	0.016	51	20

Table 7.2: BLAST vs. standard meta-blocking techniques.

useful to directly compare blocking collections that present different values of both precision and recall. To compare the scalability of the analyzed approaches I consider the overhead time t_o and the resolution time t_r, namely, the time to produce the final blocking collection and the time to execute the actual comparison respectively. Once the final block collection is built, entity profiles are compared with the Jaccard similarity to assess t_r, as in [Pap+14]; even though t_r tightly depends on the technique employed to perform the actual comparison, and its choice is an orthogonal problem not tackled here. In fact, it is important to notice that the high t_o of meta-blocking approaches becomes insignificant, when advanced, time-consuming entity matching methods are employed [PPK14].

I implemented BLAST in `Java 8` as extension of the open source framework presented in [Pap+15]. The experiments have been performed under `Ubuntu 14.04`, with 40GB of ram, and `Intel Xeon E5-2670v2` 2.50 GHz.

7.1.2 High Quality Blocking

In Table 7.2 I present the performance of BLAST compared with traditional meta-blocking. The baseline is Token Blocking in combination with traditional meta-blocking [Pap+14] (first line). To demonstrate that traditional meta-blocking can-

not fully take advantage of loosely schema-aware blocking, I also compare the naïve combination of AC and unsupervised meta-blocking (second row). I also compare BLAST against supervised meta-blocking [PPK14], using as training set the 10% of the entity profiles matched in the ground truth (third line).

For unsupervised meta-blocking, I choose WNP as pruning schema; PC and PQ are obtained as the average of the PC and PQ values generated with all possible weighting schema of traditional meta-blocking [Pap+14]. I choose WNP because is the most suitable pruning schema for maximizing PC; in fact, cardinality-based pruning cannot achieve the same level of recall [PPK14].

For supervised meta-blocking I employ WEP schema in combination with Support Vector Machine (SVM), since it is the classification algorithm that, in average, has the best F_1 score. The choice of WEP is due to incompatibility of WNP with supervised meta-blocking, since it always selects a global optimum threshold [PPK14]; nevertheless weight-based pruning remain the best choice to maximize PC.

For BLAST I employ LMI and AM3. LSH-based attribute-match induction is used only for `dbp` in both BLAST and unsupervised meta-blocking, since it yields almost identical results, but in significantly lower time, and both the approaches can benefit from its employment. Finally, for all the experiments, Token Blocking is applied in conjunction to block purging [Pap+13], a post-processing step that given a blocking collection remove oversized block, that correspond to highly frequent terms[8]. For all the approaches that require a attribute-match induction phase, t_o includes both the time of LMI (or AC) and Token Blocking.

PC and PQ analysis: Compared to the baseline, BLAST can achieve significantly higher PQ, barely affecting PC (Figure 7.1 a).

In fact, the $\Delta PC(\mathcal{B}, \mathcal{B}')$ is above -5% for all the datasets; in the case of `pr1` PC is even enhanced. Interestingly, partially mappable comparisons can benefit the most: for `mv` and `dbp` PQ increases of two order of magnitude, with respect to the baseline method, and compared with the supervised meta-blocking, the increase is of 1,300% and 4,300% respectively, maintaining almost the same PC.

[8]This step require a parameter setting to retain blocks; I tune block purging to achieve the best F_1 score for the baseline approach, guaranteeing a $\Delta PC < 3\%$; due to this step results can be slightly different, though not worse, from those presented in [PPK14]

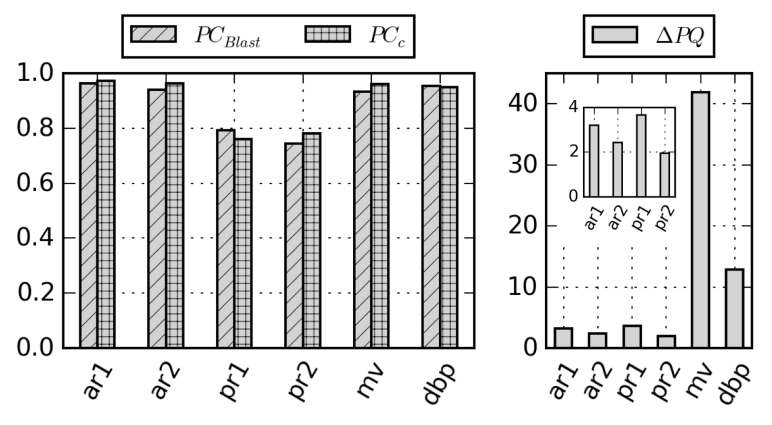


Figure 7.1: PC comparison and ΔPQ between BLAST and standard meta-blocking.

Another observation is about the inefficacy of AC as preprocessing step for Token Blocking and meta-blocking (line 2 of Table 7.2). In fact, for all the comparisons the values of PC and PQ results almost unchanged compared to the baseline. The only exceptions are `pr1`, where PC is significantly increased, and `dbp`, where PC is slightly better than the other techniques, but comes with an extremely low PQ.

7.1.3 Entropy Experiments

The results in Table 7.2 show how BLAST can actually take full advantage of attribute-match induction techniques. To deeply analize the contribution of the aggregate entropy employed to normalize the χ^2 score, I compare my technique running BLAST with and without considering the entropy contribution. The result of this comparison is shown in Figure 7.2, obtained applying $AM3$ as threshold selection scheme. The figure shows how PC remains almost the same; while PQ significantly increase for partially-mappable datasets, and less markedly for fully-mappable ones.

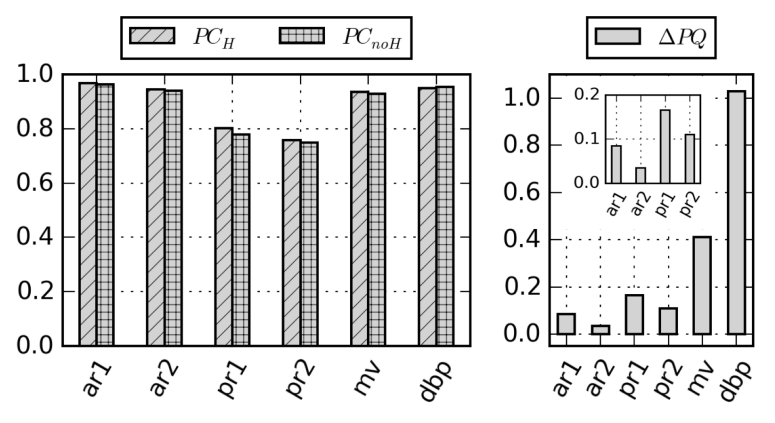


Figure 7.2: PC comparison and ΔPQ between BLAST with and without considering cluster entropies.

7.1.4 Attribute-match Induction: LMI vs. AC

BLAST supports both LMI and AC as attribute-match induction techniques. Figure 7.3 reports the comparison of BLAST with LMI and BLAST with AC. For the datasets `ar1`, `pr1` and `mv` LMI and AC identify the same clusters, therefore the resulting PC and PQ are identical; but for the other datasets LMI performs better. The reason behind this is that LMI tries to produce cohesive cluster of attributes (i.e., all attribute in the cluster are all highly similar to each other); while AC aims to group together attributes similar to other similar attributes (i.e., each grouped attribute has at least one highly similar attribute in its cluster).

7.1.5 Threshold Schemes: AM2 vs. AM3

In this experiment I evaluated the performance of the two different threshold selection schemes (see section 5.3.2). As shown in Figure 7.4 AM3 performs slightly better than AM2 in average.

7.1.6 LSH-based Attribute-Match Induction

Table 7.3 lists the execution times of different configurations of LMI[9]. The first column refers to LMI applied without LSH; the remaining columns list the exe-

[9] AC is not reported, but, in terms of execution time, it leads to almost identical results.

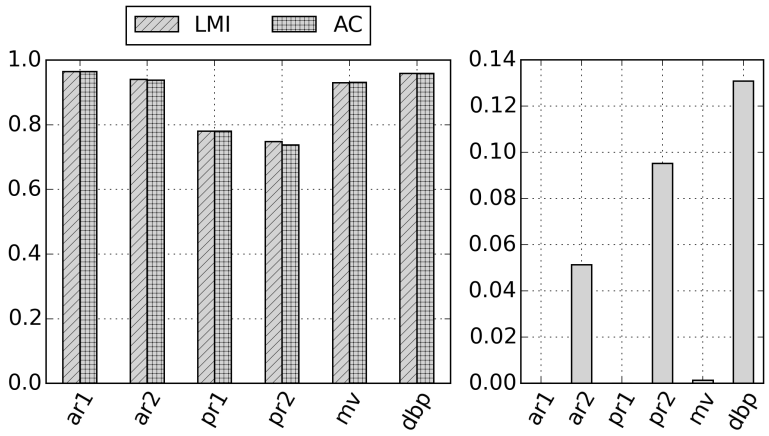


Figure 7.3: PC comparison and ΔPQ between BLAST with LMI and AC.

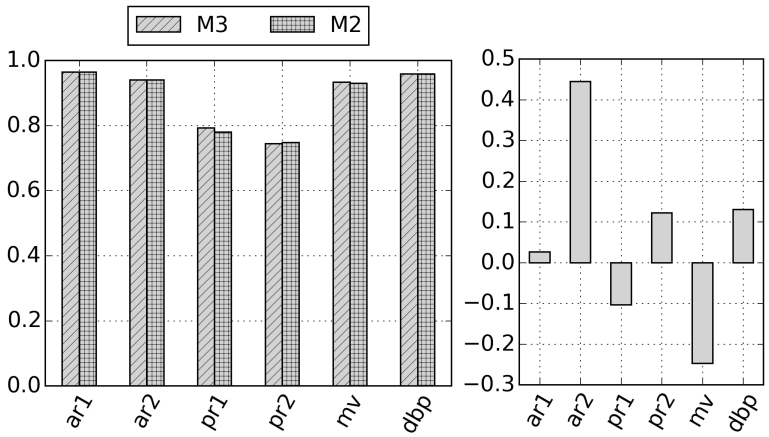


Figure 7.4: PC comparison and ΔPQ between BLAST with M2 and M3.

cution time of LMI applied in combination to LSH with several parametrization, i.e., with different number of rows and bands (the subscripts correspond to the estimated threshold of LSH).

LMI w/o LSH	LMI + $LSH_{.10}$	LMI + $LSH_{.22}$	LMI + $LSH_{.32}$	LMI + $LSH_{.41}$	LMI + $LSH_{.55}$	LMI + $LSH_{.64}$
12.5 h	1.9 h	1.5 h	1.3 h	1.2 h	0.9 h	0.7 h

Table 7.3: LMI execution time.

Figure 7.5 shows how LSH affects the final results of BLAST combined with LMI in terms of PC. Basically, up to a threshold value of .35 (i.e., Jaccard similarity equals to .35), the PC is not affected ($PC = 99.99\%$), meaning that all the

matching profile pairs are successfully indexed in the blocking collection. PQ is not reported, but for the points where $PC = 99.99\%$ is identical, i.e., it is not affected by the LSH threshold. For threshold greater than $.35$, on the contrary, the techniques starts to fail indexing some profile pairs, entailing a degradation of the final result. In other words, for thresholds that exclude to many attribute comparisons, LMI fails to recognize similar attributes and produces incomplete cluster of attributes. Nevertheless, I notice that even for conservative threshold (e.g. $.10$), the execution time is significantly enhanced.

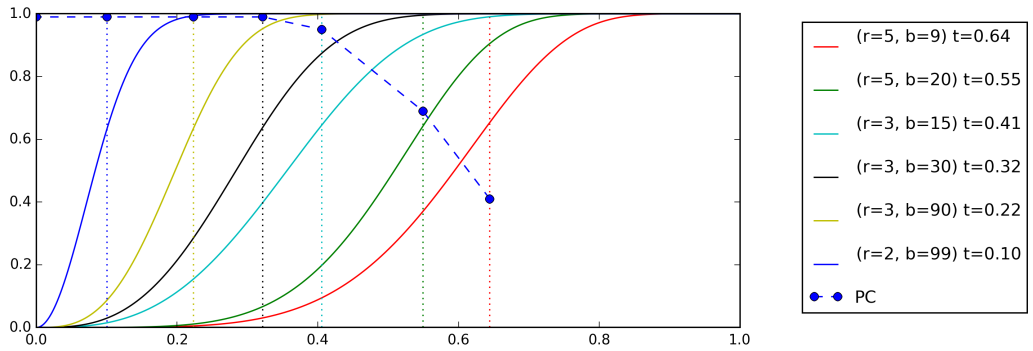


Figure 7.5: PCs with different LSH configurations in combination with LMI.

7.2 WHATSIT evaluation

The experiments proposed in this section aim to evaluate three main aspects of the approach, and, in particular: (1) the extent in which entropy-based measures are able to identify the topics described by data source properties (see Section 7.2.1); (2) the effectiveness of the signature in representing and recognizing concepts in data sources (see Section 7.2.2); (3) the effectiveness of the matching algorithm introduced in finding close signatures (see Section 7.2.3). In all the experiments, the re-weighted and pseudo additive entropies have been computed with the values of a and b equal to 1.5 and 0.5 respectively. I empirically discovered that these values typically provide good results in all the dataset considered. All the experiments had been conducted on a `m3.2xlarge AWS ec2` instance, with 8 vCPU and 30GB of RAM.

The reference ontology. DBpedia (version 3.9) has been adopted in my experiments as reference ontology. It conceptualizes the real world through a hierarchy structure made of 610 classes as described in the DBpedia website[10]. Each class comprises a rich set of datatype and object properties (e.g., the class Person includes more than 3k properties), and a large number of instances is provided for most of the classes (e.g., there are more than 760k instances belonging to the class Person in the English version, more than 300k belonging to the class Work). In my experiments, I considered only the DBpedia properties containing a sufficient number of instances and unique values to compute meaningful entropy values. In particular, I considered properties with at least 100 elements and assuming at least 5 different values. Moreover, I applied a stop-word list of terms to discard properties recording meta-information about how the class is coded in DBpedia (e.g., I did not consider properties with names containing one of the following prefixes: rdf, owl, uri, wiki, thumbnail, alias, label, etc.). This way, I remove noise generated by "system" properties which do not convey any semantics about what the data is describing.

7.2.1 Experimenting entropy as semantic identifier of a property subject

The goal of the experiments described in this section is to show that the entropy effectively identifies property topics and does not depend on the "actual" values assumed by a property in a specific data source. In other words, I performed a number of experiments to show that properties representing the same topic in different data sources have close entropy values.

First of all, I want to show that my implementations of the entropy-based measures are not affected by the number of instances available in the specific property. For this reason, I analyzed the entropy of samples of DBpedia properties with different dimensions. My claim is that fixed the number of instances taken into account, the entropy values computed are close in all the samples.

Table 7.4 summarizes the results of my experiments[11]. Column 2 describes

[10] http://wiki.dbpedia.org/Datasets/DatasetStatistics

[11] Even if the table shows the analysis performed on only few properties belonging to three

			Dimension of the subsets			
Element	Instances	Measure	10%	30%	50%	90%
Cl.: Artist	56431	CLA (var)	2.31×10^{-5}	7.27×10^{-6}	3.8×10^{-6}	1.64×10^{-6}
Prop.:	c. 23667	CLA (mean)	0.6418	0.6196	0.6095	0.5991
birthPlace		WEI (var)	2.14×10^{-6}	1.01×10^{-6}	6.51×10^{-7}	3.47×10^{-7}
		WEI (mean)	0.1392	0.1231	0.1175	0.1127
		PAE (var)	2.32×10^{-5}	6.45×10^{-6}	4.04×10^{-6}	1.88×10^{-6}
		PAE (mean)	0.7231	0.7248	0.7243	0.7252
Cl.: Artist	18966	CLA (var)	1.81×10^{-5}	3.3×10^{-6}	3.03×10^{-6}	9.13×10^{-7}
Prop.:	c. 895	CLA (mean)	0.2165	0.2006	0.1934	0.1873
nationality		WEI (var)	1.45×10^{-6}	1.06×10^{-6}	3.23×10^{-7}	2.48×10^{-7}
		WEI (mean)	0.0592	0.0534	0.0517	0.0499
		PAE (var)	1.82×10^{-5}	6.43×10^{-6}	4×10^{-6}	1.97×10^{-6}
		PAE (mean)	0.2567	0.2567	0.2564	0.2564
Cl.: Writer	15498	CLA (var)	8.25×10^{-5}	1.79×10^{-5}	1.36×10^{-5}	7.09×10^{-6}
Prop.:	c. 9303	CLA (mean)	0.6824	0.6567	0.6487	0.6400
birthPlace		WEI (var)	2.04×10^{-5}	7.87×10^{-6}	3.34×10^{-6}	$.06\times10^{-6}$
		WEI (mean)	0.1766	0.1488	0.1396	0.1324
birthPlace		PAE (var)	7.45×10^{-5}	2.22×10^{-5}	2.18×10^{-5}	6.56×10^{-6}
		PAE (mean)	0.7277	0.7319	0.7329	0.734
Cl.: Writer	9455	CLA (var)	6.13×10^{-5}	1.2×10^{-5}	1.36×10^{-5}	5.02×10^{-6}
Prop.:	c. 561	CLA (mean)	0.2563	0.2277	0.2168	0.2128
nationality		WEI (var)	1.93×10^{-5}	5.58×10^{-6}	2.69×10^{-6}	1.24×10^{-6}
		WEI (mean)	0.0400	0.0261	0.0215	0.0179
		PAE (var)	7.51×10^{-5}	3.92×10^{-5}	1.73×10^{-5}	9.98×10^{-6}
		PAE (mean)	0.2160	0.2205	0.2202	0.2197
Cl.: Automobile	5121	CLA (var)	1.20×10^{-4}	5.68×10^{-5}	4.38×10^{-5}	1.68×10^{-5}
Prop.:	c. 778	CLA (mean)	0.6584	0.6175	0.6015	0.5862
manufacturer		WEI (var)	1.61×10^{-4}	6.09×10^{-5}	2.88×10^{-5}	1.62×10^{-5}
		WEI (mean)	0.2786	0.2456	0.2332	0.2249
		PAE (var)	1.44×10^{-4}	4.62×10^{-5}	3.64×10^{-5}	3.31×10^{-5}
		PAE (mean)	0.6860	0.6924	0.6929	0.6924
Cl.: Automobile	4740	CLA (var)	1.77×10^{-4}	7.43×10^{-5}	4.78×10^{-5}	2.87×10^{-5}
Prop.:	c. 2403	CLA (mean)	0.6204	0.6165	0.5772	0.5652
transmission		WEI (var)	1.48×10^{-4}	2.25×10^{-5}	1.62×10^{-5}	9.15×10^{-6}
		WEI (mean)	0.2105	0.1753	0.165	0.1544
		PAE (var)	1.92×10^{-4}	6.92×10^{-5}	5.26×10^{-5}	2.18×10^{-5}
		PAE (mean)	0.6598	0.6633	0.6630	0.6653

Table 7.4: Variance and mean computation of the entropy-based measures in subsets of properties with homogeneous dimensions (CLA=Shannon entropy; WEI= re-weighted entropy; PAE = Pseudo-additive entropy).

the number and the cardinality of the instances of the property shown in Column

classes, I performed the experiment over 50+ properties belonging to 10+ classes obtaining results entirely similar to the one shown.

1. The number of instances available in the properties represented in the table ranges from 4740 to 56431. Columns from 4 to 7 show the variance of the entropy measures computed against 50 different random samples of the property with homogeneous dimensions. In particular, Column 4 shows the entropy variance of 50 random samples having a dimension equal to 10% of the whole property. This means that, for example, the first row shows the variance of 50 random subsets each one containing 5643 elements of the property *birthPlace* belonging to the class *Artist*. Columns 5-7 show the results obtained by the application of the same operation to 50 random subsets with dimension equal to 30%, 50%, and 90% of the number of instances in the property.

The results of the experiment show that the variance is typically low, thus meaning that the entropy values are close and independent of the values randomly selected in the subsets. Moreover, the variance decreases with the increasing of the number of instances taken into account. The more instances are taken into account, the more the entropy values converge to a fixed value.

In my second experiment, I want to show that entropy acts as a semantic identifier. For this reason, I compared the entropy of a property with the ones of random samples with different dimensions. Since I am comparing items of the same property, I am expecting to obtain close values. Note that entropy is sensitive to the cardinality of the property (see Section 6.1 and the results of the previous experiment) and the cardinality of a property containing a large number of instances is expected to be higher than the one with a small number of elements (see Figure 4.2). Even if the normalization makes entropy values comparable, I cannot directly compare the entropy values of samples with the whole property population , since the measurements can be affected by error due to random sampling. To propose a fair evaluation, I introduced and analyzed 95% confidence intervals. In particular, for each property, I created 50 samples having each one dimension equal to 10%, 30%, 50% of the whole number of instances. For each "dimension", I computed the entropy for all the samples, I analyzed the median value and its 95% confidence interval. Finally, I checked if the "actual" entropy value (the one computed on the all instances of the property) is contained in the confidence interval.

Class	# of Properties	Shannon Entropy			Re-Weighted Entropy			Pseudo-additive Entropy		
		10%	30%	50%	10%	30%	50%	10%	30%	50%
Actor	17	6%	88%	94%	24%	53%	71%	100%	100%	100%
Airline	12	8%	100%	100%	25%	59%	67%	100%	100%	100%
Artist	5	0%	0%	0%	0%	0%	0%	100%	100%	100%
Autom.	5	20%	60%	80%	40%	40%	80%	100%	100%	100%
Band	17	12%	59%	76%	29%	29%	59%	94%	100%	100%
Beverage	3	33%	67%	100%	33%	67%	100%	100%	100%	100%
Hockey Team	5	0%	100%	100%	60%	80%	80%	100%	100%	100%
Game	4	0%	100%	100%	25%	50%	100%	100%	100%	100%
Musical Artist	17	0%	12%	35%	18%	18%	41%	94%	100%	100%
Painter	11	0%	82%	100%	64%	91%	100%	100%	100%	100%
Politician	46	2%	54%	70%	26%	33%	50%	98%	98%	100%
Rugby Club	5	0%	100%	100%	60%	60%	60%	100%	100%	100%
Scientist	27	0%	67%	78%	22%	41%	59%	100%	100%	100%
Soccer league	5	20%	40%	60%	40%	80%	100%	80%	100%	100%
Writer	5	20%	60%	80%	40%	40%	80%	100%	100%	100%

Table 7.5: Analysis of the confidence intervals: the percentages refer to the properties that correctly represent the actual entropy value.

Table 7.5[12] shows the results of my evaluation. I observe that the Shannon entropy suffers from high bias, in particular in small datasets, since a small number of properties are within the confidence intervals (see for example the evaluation concerning 10% of the instances where for several classes – Artist, Game, Rugby, ...– no entropy value is within the confidence intervals). Conversely, the pseudo-additive accurately works well, being able to correctly approximate the actual entropy value in almost all the cases.

The comparison of DBpedia properties with their small samples guarantees that I am evaluating elements which are describing the same topic (I assume that all the instances of a property describe a feature related to the specific property represented). Moreover, the large number of instances in the properties assures that I am not comparing properties with precise "copies" of them. Nevertheless, to have a more extensive evaluation, I considered 17825 randomly selected properties both available in 2 snapshots of DBpedia, referring to the years 2007 and

[12]For sake of semplicity, the Table shows the analysis performed on only few classes. Nevertheless I performed the experiment over 50+ classes and the results showed trends similar to the ones represented.

2009 (the first containing an overall of 1.19M instances, the second 2.16M instances). It is important to note that DBPedia has evolved to such an extent that a mere 23.67% of all property-value pairs and 48.62% of the attribute names is common among both versions. For each kind of entropy and for each property, I computed the difference (normalized) of the values obtained in the two snapshots. Finally, I analyzed these values by calculating the mean, median and standard de as reported in Table 7.6.

	Shannon Entropy	Re-Weighted Entropy	Pseudo-additive Entropy
mean	0.076348	0.076348	0.069418
median	0.039265	0.029197	0.022299
std	0.098645	0.133649	0.069418

Table 7.6: Analysis of the difference of entropies computed on 17825 properties taken from 2 DBpedia snapshots.

All the distributions are right skewed, and, show a large number of values close to zero. This means that there is a big amount of properties in the snapshots having similar values of Entropy. Note that the Pseudo-additive Entropy performs better since it is able "to eliminate" more occurrences with a high difference value (see Figure 7.6, where the distributions of the Entropy values are shown).

Finally, I evaluated the behaviour of the entropy measures on different data sources describing the same topics. For this purpose, I performed an experiment with the benchmark proposed in [KTR10]. This benchmark is conceived for the evaluation of entity resolution approaches. It is composed of four collections, each one containing two datasets about the same domain (i.e., bibliographic and e-commerce) as shown in Table 7.7. The datasets describe a number of common (i.e., the same item is represented in both the sources) and different items as reported respectively in columns "Comm" and "Diff". So, for example, the first row shows that the first collection includes datasets extracted from the DBLP and ACM databases containing 2224 items which are represented in both the sources. For each attribute in the dataset, the values of Shannon Entropy, Re-Weighted Entropy, and Pseudo-additive Entropy have been computed. The Table reports the normalized difference of the entropies for the properties common in both the datasets.

Domain	Sources	Comm	Diff	Shannon	Re-Weighted	Pseudo-add.
Bibliographic	DBLP ACM	2224	463	venue: 5.88E-02 year: 2.13E-02 title: 5.44E-03 authors: 2.48E-03	4.20E-02 2.51E-02 3.40E-02 2.36E-02	2.08E-02 2.84E-03 7.91E-05 5.65E-04
Bibliographic	DBLP Scholar	5347	58903	venue: 2.50 year: 1.39E-01 title: 1.02E-02 authors: 1.06E-02	1.76 7.01E-02 .46E-02 3.43E-02	9.20E-01 1.76E-01 1.85E-02 1.99E-02
E-commerce	Amazon Google	1300	1989	price: 4.63E-01 descr: 2.45E-02 name.: 9.60E-03 manuf: 5.22E-02	6.88E-01 7.83E-02 1.98E-02 7.68E-02	1.67E-01 6.21E-03 8.27E-03 1.30E-01
E-commerce	Abt Buy	1081	16	price: 2.63E-01 name: 3.06E-03 descr.: 3.62E-02	3.82E-01 5.83E-03 5.91E-02	7.27E-02 4.74E-04 1.57E-02

Table 7.7: Analysis of the difference of entropies computed on attributes of different data sources in the same domain.

This experiment shows that the properties describing the same quality (e.g., venue, year, title, ...) in different data sources have similar entropy values (the differences are close to 0 in most of the cases). Moreover, in these datasets, the pseudo-additive entropy performs better than the other measures, thus confirming the evaluation results achieved in the previous experiments.

7.2.2 Experimenting signatures

The goal of this evaluation is to show that signatures effectively represent the data source topics. For reaching this purpose, I performed three experiments with DBpedia classes and I tested if: 1) Casual partitions of the instances related to the same class provide similar signatures; 2) The signatures of a class and the one of its superclass are close; 3) The signatures of two not related classes are different. I started the experiment by selecting three classes from DBpedia (*Writer, Artist, Automobile*) and building their signatures as shown in Figure 7.7. The first signature represents a fragment of the DBpedia Writer class, including only five representative properties for simplicity. The second describes the Artist class, i.e. the superclass of Writer. Note that the classes share properties having the same name, but since representing different entities, the values of entropy and mutual

information are different. Finally, the third signature represents five properties of the Automobile class. In Figure 7.7, I show the values of the pseudo-additive entropy (on the nodes) and Mutual Information (on the edges).

The WHATSIT technique relies on the specific contribution provided by entropy and mutual information alone. For this reason, I performed separate evaluations, by considering firstly only the nodes (thus measuring the contribution of the entropy) and secondly the edges (thus measuring the contribution of the mutual information). I adopted a Euclidean distance-based metric as, in [KN03], defined as follows. Let A and B be two equal size dependency graphs and a_i, b_j the entropy of the node i and j in graph A and B, respectively. Let m be an index that maps a node in graph A into the matching node in graph B (i.e., m(node in A) = matching node in B). The distance metric based on entropy for graph A and B is:

$D = \sqrt{\sum_i (a_i - b_{m(i)})^2}$

An analogous distance measure can be easily defined by considering mutual information instead of entropy. The result of my experiment is shown in Table 7.8, where Rows 1-3 compare signatures obtained by random equal-size partitions of the instances of the class Writer, Artist and Automobile (actually, the result shown is the mean of the distance measures obtained evaluating 10 random partitions). Rows 4-5 show the distances between the signature of the concept Writer and its superset Artist (with correct and random matches between the properties). Rows 6-7 show the distances between the previous concepts (Writer and Artist) and the concept Automobile.

#	Comparison	Distance (H - Nodes)	Distance (MI - Edges)
1	Artist - Artist	0.004	0.873
2	Writer - Writer	0.007	0.142
3	Automobile - Automobile	0.004	0.349
4	Artist - Writer (best matches)	0.346	4.531
5	Artist - Writer (random matches)	0.522	5.947
6	Artist - Automobile (best matches)	0.803	8.205
7	Writer - Automobile (best matches)	0.702	8.348

Table 7.8: Evaluation of the signatures.

The results show that both the measures detect signatures representing similar

and different concepts. As in [KN03], my experiment shows that the entropy alone provides a good account of the similarities between the classes. Nevertheless, the value of mutual information can support the decision about the closeness of two classes. Note that, as show in Table 7.9, in this experiment I selected large and high cardinality properties. This fact, let us generalize the results observed in this fragment of DBpedia.

Class	Property	# instances	Cardinality
Artist	deathPlace	16372	285
	birthPlace	56431	439
	genre	41991	1409
	nationality)	18966	6202
	occupation	35727	2437
Writer	deathPlace	6375	84
	birthPlace	15498	157
	genre	5452	326
	nationality)	9455	3796
	occupation	7585	858
Automobile	manufacturer	5121	277
	modelStart Year	1152	43
	modelEnd Year	33	0.349
	transmission)	4740	250
	designer	1167	69

Table 7.9: Description of the classes/properties involved in the experiment.

7.2.3 Matching algorithm evaluation

The goal of the experiments described in this section is to demonstrate the efficiency of the matching algorithm implemented in WHATSIT. As a first experiment, I evaluated the algorithm with randomly selected DBpedia classes as shown in Table 7.10. Firstly, I selected the target classes, i.e., the classes that I have "to discover". To make the experiment more challenging and the dataset closer to real world data, I considered, for each property, a sample with 20% of the instances available in the property. Then, I considered other DBpedia classes, both in the same IS-A hierarchy and casually selected, and I evaluated their matching. As shown in the Table, in most of the cases, the computation of the coverage

is enough to select the best DBpedia class to be associated to the input source. Only in case of tie, the computation of the mutual information is needed for the disambiguation.

Moreover, I performed a second experiment by analyzing a real movie database[13]. It originally consists of a single class with 12 properties, but, for the experiment purposes, only three properties (the ones satisfying the constraints of minimum number of instances and minimal cardinality introduced at the beginning of Section 6.2), have been considered: *director, releaseDate* and *length*. The result is a target data source containing 1406 instances. I am expecting this class to be matched with the corresponding *Film* class in DBpedia which is composed of 71629 instances.

Table 7.11, where the left part shows the input class properties and the right part the DBpedia corresponding properties, reports the results of this experiment. The first three rows show that WHATSIT finds the correct associations between the properties in the selected database and DBpedia. Note that the entropy values are close (in the confidence intervals) and relate meaningful properties. The last three rows shows, as an example, the values of three properties of another randomly selected class. The entropy values are not close, thus meaning that the source is not describing the showed properties of a Painter.

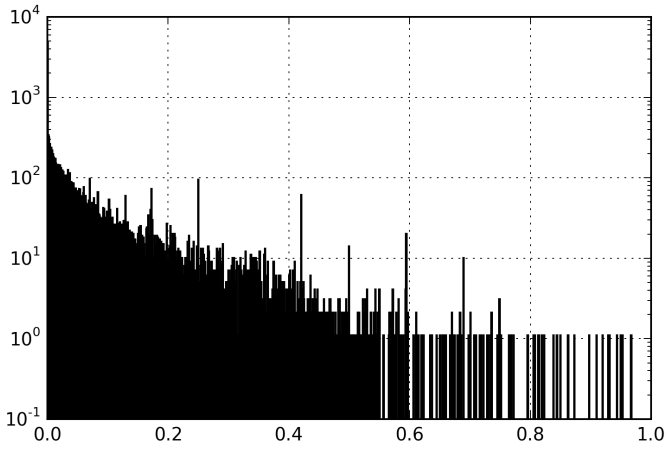
[13]http://perso.telecom-paristech.fr/ eagan/class/as2013/inf229/labs/datasets

Target Class (number of prop.)	DBpedia Class	Coverage	Coverage normalized
Beverage (3)	BaseballPlayer	3/3	1
	Beverage*	3/3	1
	BasketPlayer	2/3	0.67
Celebrity (9)	**Celebrity**	9/9	1
	Writer	7/9	0.78
	Cleri	7/9	0.78
	FootballPlayer	6/9	0.67
	Model	4/9	0.44
ChessPlayer (9)	**ChessPlayer***	8/9	0.89
	Politician	8/9	0.89
	Writer	7/9	0.78
Criminal (8)	**Criminal**	7/8	0.87
	BaseballPlayer	4/8	0.5
	Cleri	4/8	0.5
Film (5)	**Film**	5/5	1
	MusicalWork	2/5	0.4
MotorRacer (17)	**MotorRacer**	17/17	1
	Cleri	14/17	0.82
	Politician	14/17	0.82
	Actor	12/17	0.71
	Scientist	9/17	0.53
Painter (11)	**Painter**	11/11	1
	Actor	7/11	0.64
	Writer	7/11	0.64
	Airline	6/11	0.54
	BasketballPlayer	5/11	0.54

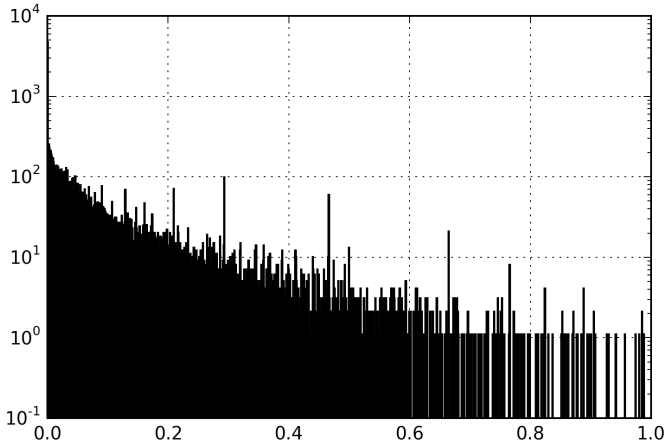
Table 7.10: Subsets (20%) of the instances of a DBpedia classes are considered as target classes. WHATSIT selected matching classes have bold font. Matching classes that require mutual information to be detected are starred.

Target Data Source			DBpedia		
Target Class Property	H^*_{target}	σ_{target}	Matching Property:{Class}	H^*_{match}	σ_{match}
director	.901	.003	director:{Film}	.9004	.0012
year	.830	.002	releaseDate:{Film}	.8301	.0007
length	.88570	.00013	runtime:{Film}	.88491	.00003
			birthPlace:{Painter}	.4277	.0013
			birthYear:{Painter}	.8402	.0057
			country:{Painter}	.1516	.006

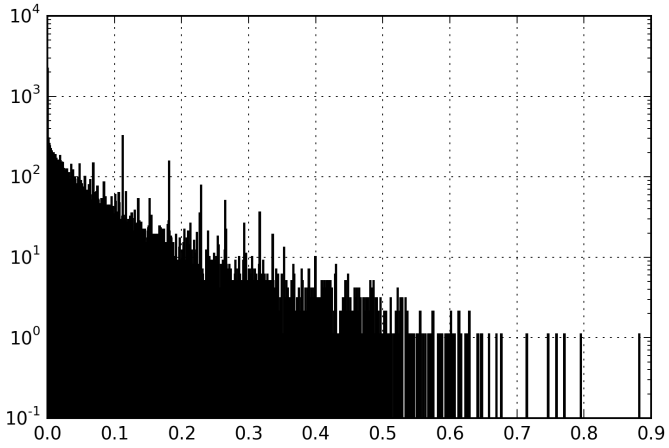
Table 7.11: Matching a movie dataset. The star entropy H^* means normalized entropy. σ is the standard deviation.



(a) Shannon Entropy



(b) Re-Weighted Entropy



(c) Pseudo-additive Entropy

Figure 7.6: Frequency distribution of the difference of entropies computed on 17825 properties taken from 2 DBpedia snapshots.

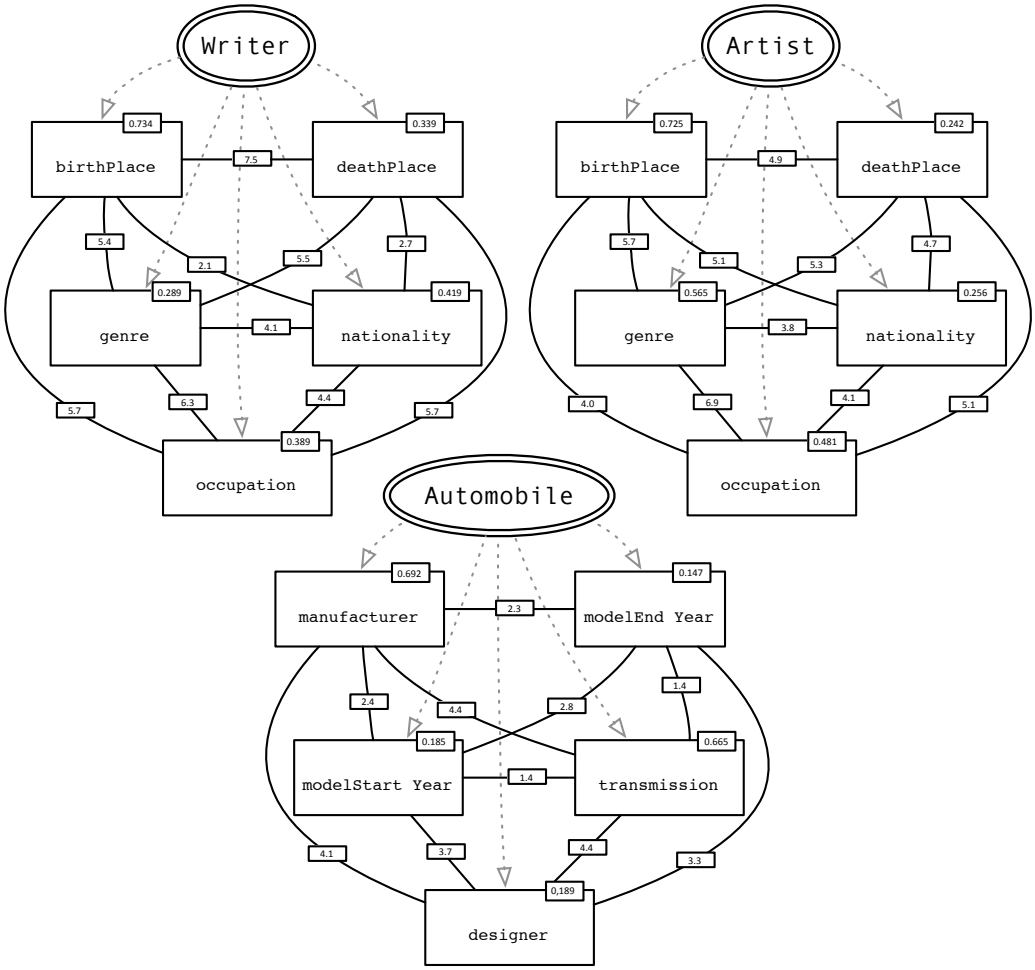


Figure 7.7: The signatures of three DBpedia classes. The values in the boxes are pseudo-additive values for entropy and mutual information.

Chapter 8

Conclusions and Future Work

Conclusions. In this thesis I demonstrated how *loose schema information* can be extracted and exploited to overcome the impossibility of achieve a schema-alignment when the data source are chatacterized by high heterogeneity, volume and noise, such as the data of the Web.

I employed *loosely schema-aware* techniques to support blocking for Entity Resolution, and to identify the topics of a data source given a reference vocabulary. For the former, I demonstrated how loose schema information can significantly increase the quality of meta-blocking, the state of the art technique to enhance the quality of a blocking collection. For the latter, I defined the task of identifying the topics of a data source given a reference vocabulary as a signature matching problem, and I demonstrated how loose schema information can be actually employed to determinate the signatures of target data sources and reference vocabularies.

I implemented two prototypal systems of my proposed approaches: BLAST (Blocking with Loosely-Aware Schema Techniques), that holistically merge loosely schema-awareblocking and meta-blocking; and WHATSIT a system that employs loosely schema-awaretechniques to build the signatures of both a reference vocabulary (I employed DBpedia) and the target data sources, to identify the topics of these latter.

Finally, I experimentally evaluated the proposed approaches, demonstrating how BLAST can outperform the state-of-the-art meta-blocking approaches, and

how WHATSIT can actually be employed to detect topics of a data source.

Future Work. My ongoing research is focused on the application of BLAST for the so called *Big Data* [DS15]. In particular for data integration and data mining applications [Ben+01; Ber+; GSV15; SG14; BGS13], and exploratory data analysis [SZ15; BSZ].

I am also investigating how to adapt my technique to parallel distributed systems [LRU14; ISB14], such as `Spark` [Zah+12] and `Flink` [Ale+14], to achieve a faster execution time. In particular, with graph-parallel computation [Mal+09; Xin+13] novel graph based meta-blocking approaches could be studied, taking advantage of the iterative computation, not practicable with the centralized computation, due to the huge volume of the data involved.

Publications Related to This Thesis

[Ber+] Sonia Bergamaschi, Davide Ferrari, Francesco Guerra, Giovanni Simonini, and Yannis Velegrakis. "A statistical approach for discovering the topics of a data source". In: *Journal of Data Semantics*.

[Ber+14] Sonia Bergamaschi, Davide Ferrari, Francesco Guerra, and Giovanni Simonini. "Discovering the topics of a data source: a statistical approach". In: *Surfacing the Deep and the Social Web (SDSW) Workshop held at International Semantic Web Conference*. 2014.

[BGS13] Sonia Bergamaschi, Francesco Guerra, and Giovanni Simonini. "Keyword Search over Relational Databases: Issues, Approaches and Open Challenges". In: *Bridging Between Information Retrieval and Databases - PROMISE Winter School 2013, Bressanone, Italy, February 4-8, 2013. Revised Tutorial Lectures*. 2013, pp. 54–73. DOI: 10.1007/978-3-642-54798-0_3. URL: http://dx.doi.org/10.1007/978-3-642-54798-0_3.

[BSZ] Sonia Bergamaschi, Giovanni Simonini, and Song Zhu. "Enhancing big data exploration with faceted browsing". In: *CLADAG 2015, Cagliary, Italy 2015*.

[GSV15] Francesco Guerra, Giovanni Simonini, and Maurizio Vincini. "Supporting Image Search with Tag Clouds: A Preliminary Approach". In: *Adv. in MM* 2015 (2015), 439020:1–439020:10. DOI: 10.1155/2015/439020. URL: http://dx.doi.org/10.1155/2015/439020.

[ISB14] Matteo Interlandi, Giovanni Simonini, and Sonia Bergamaschi. "Towards Declarative Imperative Data-parallel Systems". In: *22nd Italian Symposium on Advanced Database Systems, SEBD 2014, Sorrento Coast, Italy, June 16-18, 2014*. 2014, pp. 97–104.

[SG14] Giovanni Simonini and Francesco Guerra. "Using big data to support automatic Word Sense Disambiguation". In: *International Conference on High Performance Computing & Simulation, HPCS 2014,*

Bologna, Italy, 21-25 July, 2014. 2014, pp. 311–314. DOI: 10.1109/HPCSim.2014.6903701. URL: http://dx.doi.org/10.1109/HPCSim.2014.6903701.

[SZ15] Giovanni Simonini and Song Zhu. "Big data exploration with faceted browsing". In: *2015 International Conference on High Performance Computing & Simulation, HPCS 2015, Amsterdam, Netherlands, July 20-24, 2015*. 2015, pp. 541–544. DOI: 10.1109/HPCSim.2015.7237087. URL: http://dx.doi.org/10.1109/HPCSim.2015.7237087.

Submitted Articles Related to This Thesis

[SBJa] Giovanni Simonini, Sonia Bergamaschi, and H.V. Jagadish. “Blast: An Entity Resolution Scheme Based on Loosely Schema-aware Blocking”. In: *SIGMOD 2016*.

[SBJb] Giovanni Simonini, Sonia Bergamaschi, and H.V. Jagadish. “Blast: Blocking with Loosely-Aware Schema Techniques”. In: *VLDB 2016*.

Bibliography

[AK11] Alan Agresti and Maria Kateri. "Categorical Data Analysis". In: *International Encyclopedia of Statistical Science*. 2011, pp. 206–208. DOI: 10.1007/978-3-642-04898-2_161. URL: http://dx.doi.org/10.1007/978-3-642-04898-2_161.

[Ale+14] Alexander Alexandrov et al. "The Stratosphere platform for big data analytics". In: *VLDB J.* 23.6 (2014), pp. 939–964. DOI: 10.1007/s00778-014-0357-y. URL: http://dx.doi.org/10.1007/s00778-014-0357-y.

[Ben+01] Domenico Beneventano, Sonia Bergamaschi, Francesco Guerra, and Maurizio Vincini. "The MOMIS Approach to Information Integration". In: *ICEIS (1)*. 2001, pp. 194–198.

[Ber+] Sonia Bergamaschi, Davide Ferrari, Francesco Guerra, Giovanni Simonini, and Yannis Velegrakis. "A statistical approach for discovering the topics of a data source". In: *Journal of Data Semantics*.

[Ber+07] Sonia Bergamaschi, Claudio Sartori, Francesco Guerra, and Mirko Orsini. "Extracting Relevant Attribute Values for Improved Search". In: *IEEE Internet Computing* 11.5 (2007), pp. 26–35.

[Ber+14] Sonia Bergamaschi, Davide Ferrari, Francesco Guerra, and Giovanni Simonini. "Discovering the topics of a data source: a statistical approach". In: *Surfacing the Deep and the Social Web (SDSW) Workshop held at International Semantic Web Conference*. 2014.

[BGS13] Sonia Bergamaschi, Francesco Guerra, and Giovanni Simonini. "Keyword Search over Relational Databases: Issues, Approaches and Open Challenges". In: *Bridging Between Information Retrieval and Databases - PROMISE Winter School 2013, Bressanone, Italy, February 4-8, 2013. Revised Tutorial Lectures*. 2013, pp. 54–73. DOI: 10.1007/978-3-642-54798-0_3. URL: http://dx.doi.org/10.1007/978-3-642-54798-0_3.

[BHBL09] Christian Bizer, Tom Heath, and Tim Berners-Lee. "Linked data-the story so far". In: *Semantic Services, Interoperability and Web Applications: Emerging Concepts* (2009), pp. 205–227.

[Ble12] David M. Blei. "Probabilistic topic models". In: *Commun. ACM* 55.4 (2012), pp. 77–84.

[BLN86] Carlo Batini, Maurizio Lenzerini, and Shamkant B. Navathe. "A Comparative Analysis of Methodologies for Database Schema Integration". In: *ACM Comput. Surv.* 18.4 (1986), pp. 323–364. DOI: 10.1145/27633.27634. URL: http://doi.acm.org/10.1145/27633.27634.

[Bro97] Andrei Z Broder. "On the resemblance and containment of documents". In: *Compression and Complexity of Sequences 1997. Proceedings*. IEEE. 1997, pp. 21–29.

[BSZ] Sonia Bergamaschi, Giovanni Simonini, and Song Zhu. "Enhancing big data exploration with faceted browsing". In: *CLADAG 2015, Cagliary, Italy 2015*.

[Caf+08] Michael J Cafarella, Alon Halevy, Daisy Zhe Wang, Eugene Wu, and Yang Zhang. "Webtables: exploring the power of tables on the web". In: *Proceedings of the VLDB Endowment* 1.1 (2008), pp. 538–549.

[Che76] Peter P. Chen. "The Entity-Relationship Model - Toward a Unified View of Data". In: *ACM Trans. Database Syst.* 1.1 (1976), pp. 9–36. DOI: 10.1145/320434.320440. URL: http://doi.acm.org/10.1145/320434.320440.

[Chr12a] Peter Christen. "A survey of indexing techniques for scalable record linkage and deduplication". In: *Knowledge and Data Engineering, IEEE Transactions on* 24.9 (2012), pp. 1537–1555.

[Chr12b] Peter Christen. *Data Matching - Concepts and Techniques for Record Linkage, Entity Resolution, and Duplicate Detection*. Data-Centric Systems and Applications. Springer, 2012. ISBN: 978-3-642-31163-5. DOI: 10.1007/978-3-642-31164-2. URL: http://dx.doi.org/10.1007/978-3-642-31164-2.

[CP11] Eric Crestan and Patrick Pantel. "Web-scale table census and classification". In: *Proceedings of the Forth International Conference on Web Search and Web Data Mining, WSDM 2011, Hong Kong, China, February 9-12, 2011*. 2011, pp. 545–554. DOI: 10.1145/1935826.1935904. URL: http://doi.acm.org/10.1145/1935826.1935904.

[CSH06] Namyoun Choi, Il-Yeol Song, and Hyoil Han. "A survey on ontology mapping". In: *SIGMOD Record* 35.3 (2006), pp. 34–41. DOI: 10.1145/1168092.1168097. URL: http://doi.acm.org/10.1145/1168092.1168097.

[CT12] Thomas M Cover and Joy A Thomas. *Elements of information theory*. John Wiley & Sons, 2012.

[Dha13] Vasant Dhar. "Data science and prediction". In: *Commun. ACM* 56.12 (2013), pp. 64–73.

[DS15] Xin Luna Dong and Divesh Srivastava. "Big data integration". In: *Synthesis Lectures on Data Management* 7.1 (2015), pp. 1–198.

[ES13] Jérôme Euzenat and Pavel Shvaiko. *Ontology Matching, Second Edition*. Springer, 2013, pp. I–XVII, 1–511. ISBN: 978-3-642-38720-3.

[GM12] Lise Getoor and Ashwin Machanavajjhala. "Entity Resolution: Theory, Practice & Open Challenges". In: *PVLDB* 5.12 (2012), pp. 2018–2019. URL: http://vldb.org/pvldb/vol5/p2018_lisegetoor_vldb2012.pdf.

[GM13] Lise Getoor and Ashwin Machanavajjhala. "Entity resolution for big data". In: *The 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2013, Chicago, IL, USA, August 11-14, 2013*. 2013, p. 1527. DOI: 10.1145/2487575.2506179. URL: http://doi.acm.org/10.1145/2487575.2506179.

[Gra+01] Luis Gravano, Panagiotis G. Ipeirotis, H. V. Jagadish, Nick Koudas, S. Muthukrishnan, and Divesh Srivastava. "Approximate String Joins in a Database (Almost) for Free". In: *VLDB 2001, Proceedings of 27th International Conference on Very Large Data Bases, September 11-14, 2001, Roma, Italy*. 2001, pp. 491–500. URL: http://www.vldb.org/conf/2001/P491.pdf.

[GSV15] Francesco Guerra, Giovanni Simonini, and Maurizio Vincini. "Supporting Image Search with Tag Clouds: A Preliminary Approach". In: *Adv. in MM* 2015 (2015), 439020:1–439020:10. DOI: 10.1155/2015/439020. URL: http://dx.doi.org/10.1155/2015/439020.

[HC67] Jan Havrda and František Charvát. "Quantification method of classification processes. Concept of structural a-entropy". In: *Kybernetika* 3.1 (1967), pp. 30–35.

[HFJ12] Lushan Han, Tim Finin, and Anupam Joshi. “Schema-free structured querying of DBpedia data”. In: *CIKM*. Ed. by Xue wen Chen, Guy Lebanon, Haixun Wang, and Mohammed J. Zaki. ACM, 2012, pp. 2090–2093. ISBN: 978-1-4503-1156-4.

[ISB14] Matteo Interlandi, Giovanni Simonini, and Sonia Bergamaschi. “Towards Declarative Imperative Data-parallel Systems”. In: *22nd Italian Symposium on Advanced Database Systems, SEBD 2014, Sorrento Coast, Italy, June 16-18, 2014*. 2014, pp. 97–104.

[KL10] Hung-sik Kim and Dongwon Lee. “HARRA: fast iterative hashed record linkage for large-scale data collections”. In: *EDBT 2010, 13th International Conference on Extending Database Technology, Lausanne, Switzerland, March 22-26, 2010, Proceedings*. 2010, pp. 525–536. DOI: 10.1145/1739041.1739104. URL: http://doi.acm.org/10.1145/1739041.1739104.

[KN03] Jaewoo Kang and Jeffrey F. Naughton. “On Schema Matching with Opaque Column Names and Data Values”. In: *SIGMOD Conference*. Ed. by Alon Y. Halevy, Zachary G. Ives, and AnHai Doan. ACM, 2003, pp. 205–216. ISBN: 1-58113-634-X.

[KR10] Hanna Köpcke and Erhard Rahm. “Frameworks for entity matching: A comparison”. In: *Data Knowl. Eng.* 69.2 (2010), pp. 197–210. DOI: 10.1016/j.datak.2009.10.003. URL: http://dx.doi.org/10.1016/j.datak.2009.10.003.

[KTR10] Hanna Köpcke, Andreas Thor, and Erhard Rahm. “Evaluation of entity resolution approaches on real-world match problems”. In: *PVLDB* 3.1 (2010), pp. 484–493. URL: http://www.comp.nus.edu.sg/~vldb2010/proceedings/files/papers/E04.pdf.

[LRU14] Jure Leskovec, Anand Rajaraman, and Jeffrey D. Ullman. *Mining of Massive Datasets, 2nd Ed*. Cambridge University Press, 2014. ISBN: 978-1107077232.

[Mad+07] Jayant Madhavan, Shirley Cohen, Xin Luna Dong, Alon Y. Halevy, Shawn R. Jeffery, David Ko, and Cong Yu. “Web-Scale Data Integration: You can afford to Pay as You Go”. In: *CIDR 2007, Third Biennial Conference on Innovative Data Systems Research, Asilomar, CA, USA, January 7-10, 2007, Online Proceedings*. 2007, pp. 342–350. URL: http://www.cidrdb.org/cidr2007/papers/cidr07p40.pdf.

[Mad+09] Jayant Madhavan, Loredana Afanasiev, Lyublena Antova, and Alon Y. Halevy. "Harnessing the Deep Web: Present and Future". In: *CIDR*. www.cidrdb.org, 2009.

[Mal+09] Grzegorz Malewicz, Matthew H. Austern, Aart J. C. Bik, James C. Dehnert, Ilan Horn, Naty Leiser, and Grzegorz Czajkowski. "Pregel: a system for large-scale graph processing". In: *SPAA 2009: Proceedings of the 21st Annual ACM Symposium on Parallelism in Algorithms and Architectures, Calgary, Alberta, Canada, August 11-13, 2009*. 2009, p. 48. DOI: 10.1145/1583991.1584010. URL: http://doi.acm.org/10.1145/1583991.1584010.

[MNU00] Andrew McCallum, Kamal Nigam, and Lyle H. Ungar. "Efficient clustering of high-dimensional data sets with application to reference matching". In: *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining, Boston, MA, USA, August 20-23, 2000*. 2000, pp. 169–178. DOI: 10.1145/347090.347123. URL: http://doi.acm.org/10.1145/347090.347123.

[MPB14] Robert Meusel, Petar Petrovski, and Christian Bizer. "The WebDataCommons Microdata, RDFa and Microformat Dataset Series". In: *The Semantic Web - ISWC 2014 - 13th International Semantic Web Conference, Riva del Garda, Italy, October 19-23, 2014. Proceedings, Part I*. 2014, pp. 277–292. DOI: 10.1007/978-3-319-11964-9_18. URL: http://dx.doi.org/10.1007/978-3-319-11964-9_18.

[MT13a] Yongtao Ma and Thanh Tran. "TYPiMatch: type-specific unsupervised learning of keys and key values for heterogeneous web data integration". In: *Sixth ACM International Conference on Web Search and Data Mining, WSDM 2013, Rome, Italy, February 4-8, 2013*. 2013, pp. 325–334. DOI: 10.1145/2433396.2433439. URL: http://doi.acm.org/10.1145/2433396.2433439.

[MT13b] Yongtao Ma and Thanh Tran. "TYPiMatch: type-specific unsupervised learning of keys and key values for heterogeneous web data integration". In: *Sixth ACM International Conference on Web Search and Data Mining, WSDM 2013, Rome, Italy, February 4-8, 2013*. 2013, pp. 325–334. DOI: 10.1145/2433396.2433439. URL: http://doi.acm.org/10.1145/2433396.2433439.

[NH10] Felix Naumann and Melanie Herschel. *An Introduction to Duplicate Detection*. Synthesis Lectures on Data Management. Morgan & Claypool Publishers, 2010. DOI: 10.2200/

S00262ED1V01Y201003DTM003. URL: http://dx.doi.org/10.2200/S00262ED1V01Y201003DTM003.

[Ore+08] Eyal Oren, Renaud Delbru, Michele Catasta, Richard Cyganiak, Holger Stenzhorn, and Giovanni Tummarello. "Sindice.com: a document-oriented lookup index for open linked data". In: *IJMSO* 3.1 (2008), pp. 37–52.

[Pap+12] George Papadakis, Ekaterini Ioannou, Claudia Niederée, Themis Palpanas, and Wolfgang Nejdl. "Beyond 100 million entities: large-scale blocking-based resolution for heterogeneous data". In: *Proceedings of the Fifth International Conference on Web Search and Web Data Mining, WSDM 2012, Seattle, WA, USA, February 8-12, 2012*. 2012, pp. 53–62. DOI: 10.1145/2124295.2124305. URL: http://doi.acm.org/10.1145/2124295.2124305.

[Pap+13] George Papadakis, Ekaterini Ioannou, Themis Palpanas, Claudia Niederée, and Wolfgang Nejdl. "A Blocking Framework for Entity Resolution in Highly Heterogeneous Information Spaces". In: *IEEE Trans. Knowl. Data Eng.* 25.12 (2013), pp. 2665–2682. DOI: 10.1109/TKDE.2012.150. URL: http://dx.doi.org/10.1109/TKDE.2012.150.

[Pap+14] George Papadakis, Georgia Koutrika, Themis Palpanas, and Wolfgang Nejdl. "Meta-blocking: Taking entity resolutionto the next level". In: *Knowledge and Data Engineering, IEEE Transactions on* 26.8 (2014), pp. 1946–1960.

[Pap+15] George Papadakis, George Alexiou, George Papastefanatos, and Georgia Koutrika. "Schema-agnostic vs schema-based configurations for blocking methods on homogeneous data". In: *Proceedings of the VLDB Endowment* 9.4 (2015), pp. 312–323.

[PPK14] George Papadakis, George Papastefanatos, and Georgia Koutrika. "Supervised Meta-blocking". In: *PVLDB* 7.14 (2014), pp. 1929–1940. URL: http://www.vldb.org/pvldb/vol7/p1929-papadakis.pdf.

[Qiu+15] Disheng Qiu, Luciano Barbosa, Xin Luna Dong, Yanyan Shen, and Divesh Srivastava. "DEXTER: Large-Scale Discovery and Extraction of Product Specifications on the Web". In: *PVLDB* 8.13 (2015), pp. 2194–2205. URL: http://www.vldb.org/pvldb/vol8/p2194-qiu.pdf.

[Rah11] Erhard Rahm. "Towards Large-Scale Schema and Ontology Matching". In: *Schema Matching and Mapping*. 2011, pp. 3–27. DOI: 10.1007/978-3-642-16518-4_1. URL: http://dx.doi.org/10.1007/978-3-642-16518-4_1.

[RB01] E. Rahm and P. A. Bernstein. "A survey of approaches to automatic schema matching". In: *VLDB Journal* 10.4 (2001), pp. 334–350.

[SBJa] Giovanni Simonini, Sonia Bergamaschi, and H.V. Jagadish. "Blast: An Entity Resolution Scheme Based on Loosely Schema-aware Blocking". In: *SIGMOD 2016*.

[SBJb] Giovanni Simonini, Sonia Bergamaschi, and H.V. Jagadish. "Blast: Blocking with Loosely-Aware Schema Techniques". In: *VLDB 2016*.

[Sch+12] Balthasar A. C. Schopman, Shenghui Wang, Antoine Isaac, and Stefan Schlobach. "Instance-Based Ontology Matching by Instance Enrichment". In: *J. Data Semantics* 1.4 (2012), pp. 219–236. DOI: 10.1007/s13740-012-0011-z. URL: http://dx.doi.org/10.1007/s13740-012-0011-z.

[SE13] Pavel Shvaiko and Jérôme Euzenat. "Ontology Matching: State of the Art and Future Challenges". In: *IEEE Trans. Knowl. Data Eng.* 25.1 (2013), pp. 158–176. DOI: 10.1109/TKDE.2011.253. URL: http://dx.doi.org/10.1109/TKDE.2011.253.

[SG14] Giovanni Simonini and Francesco Guerra. "Using big data to support automatic Word Sense Disambiguation". In: *International Conference on High Performance Computing & Simulation, HPCS 2014, Bologna, Italy, 21-25 July, 2014*. 2014, pp. 311–314. DOI: 10.1109/HPCSim.2014.6903701. URL: http://dx.doi.org/10.1109/HPCSim.2014.6903701.

[SZ15] Giovanni Simonini and Song Zhu. "Big data exploration with faceted browsing". In: *2015 International Conference on High Performance Computing & Simulation, HPCS 2015, Amsterdam, Netherlands, July 20-24, 2015*. 2015, pp. 541–544. DOI: 10.1109/HPCSim.2015.7237087. URL: http://dx.doi.org/10.1109/HPCSim.2015.7237087.

[Tsa88] Constantino Tsallis. "Possible generalization of Boltzmann-Gibbs statistics". In: *Journal of statistical physics* 52.1-2 (1988), pp. 479–487.

[Vaa00] Aad W Van der Vaart. *Asymptotic statistics*. Vol. 3. Cambridge university press, 2000.

[Vri+11] Timothy de Vries, Hui Ke, Sanjay Chawla, and Peter Christen. "Robust Record Linkage Blocking Using Suffix Arrays and Bloom Filters". In: *TKDD* 5.2 (2011), p. 9. DOI: 10.1145/1921632.1921635. URL: http://doi.acm.org/10.1145/1921632.1921635.

[WC06] Xing Wei and W. Bruce Croft. "LDA-based document models for ad-hoc retrieval". In: *SIGIR 2006: Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Seattle, Washington, USA, August 6-11, 2006*. 2006, pp. 178–185. DOI: 10.1145/1148170.1148204. URL: http://doi.acm.org/10.1145/1148170.1148204.

[Wri08] Alex Wright. "Searching the deep web". In: *Communications of ACM* 51.10 (2008), pp. 14–15.

[Xin+13] Reynold S. Xin, Joseph E. Gonzalez, Michael J. Franklin, and Ion Stoica. "GraphX: a resilient distributed graph system on Spark". In: *First International Workshop on Graph Data Management Experiences and Systems, GRADES 2013, co-loated with SIGMOD/PODS 2013, New York, NY, USA, June 24, 2013*. 2013, p. 2. URL: http://event.cwi.nl/grades2013/02-xin.pdf.

[YJ06] Cong Yu and H. V. Jagadish. "Schema Summarization". In: *Proceedings of the 32nd International Conference on Very Large Data Bases, Seoul, Korea, September 12-15, 2006*. 2006, pp. 319–330. URL: http://dl.acm.org/citation.cfm?id=1164156.

[YPS09] Xiaoyan Yang, Cecilia M. Procopiuc, and Divesh Srivastava. "Summarizing Relational Databases". In: *PVLDB* 2.1 (2009), pp. 634–645.

[YPS11] Xiaoyan Yang, Cecilia M. Procopiuc, and Divesh Srivastava. "Summary Graphs for Relational Database Schemas". In: *PVLDB* 4.11 (2011), pp. 899–910.

[Zah+12] Matei Zaharia, Mosharaf Chowdhury, Tathagata Das, Ankur Dave, Justin Ma, Murphy McCauly, Michael J. Franklin, Scott Shenker, and Ion Stoica. "Resilient Distributed Datasets: A Fault-Tolerant Abstraction for In-Memory Cluster Computing". In: *Proceedings of the 9th USENIX Symposium on Networked Systems Design and Implementation, NSDI 2012, San Jose, CA, USA, April 25-27, 2012*. 2012, pp. 15–28. URL: https://www.usenix.org/conference/nsdi12/technical-sessions/presentation/zaharia.

[ZCQ07] Xiang Zhang, Gong Cheng, and Yuzhong Qu. "Ontology summarization based on rdf sentence graph". In: *Proceedings of the 16th International Conference on World Wide Web, WWW 2007, Banff, Alberta, Canada, May 8-12, 2007*. 2007, pp. 707–716. DOI: 10.1145/1242572.1242668. URL: http://doi.acm.org/10.1145/1242572.1242668.