Semantic Tabular Data Annotation with Knowledge Bases for Data Interoperability

by

NGUYEN Tri Phuc

Dissertation

submitted to the Department of Informatics

in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

The Graduate University for Advanced Studies, SOKENDAI

March 2020

Committee

Advisor	Dr. Hideaki Takeda Professor of National Institute of Informatics/ SOKENDAI
Subadvisor	Dr. Ryutaro Ichise Associate Professor of National Institute of Informatics/ SOKENDAI
Subadvisor	Dr. Ikki Ohmukai Associate Professor of Tokyo University
Examiner	Dr. Akiko Aizawa Professor of National Institute of Informatics/ Tokyo University / SOKENDAI
Examiner	Dr. Ryota Kobayashi Assistant Professor of National Institute of Informatics/ SOKENDAI

Acknowledgments

I would like to express my best gratitude to my advisor Prof. Hideaki Takeda for his guidance. Thank Sensei very much for your inspiration, encouragement, and patience. It is a priceless value and the excitement and productivity of my research.

I would like to thank Prof. Ryutaro Ichise for his continuous support, helpful comments on my research, as well as, allowing me to access his laboratory servers, and other resources.

I would like to thank the advisory committee Prof. Ikki Ohmukai, Prof. Akiko Aizawa, and Prof. Ryota Kobayashi for their insightful comments, suggestions on my research. It allows me to widen my research from various perspectives.

I would like to show my great respect to Dr. Khai Nguyen and Dr. Natthawut Kertkeidkachorn for mentoring and collaborating. I appreciate my lab members, NII friends for their willingness to discuss and provide thoughtful comments on my research.

Thank Tanaka-san, NII, and Daigakuin support team for technical, document support, and useful recommendation.

I gratefully acknowledge the funding sources from MEXT Honors, NII scholarship, and Takedaken that make my Ph.D. possible.

Last but not least, I am grateful to my cat, and my family for always being by my side with the constant source of love and motivation, this accomplishment would have not been possible without them.

Abstract

Thank the Open Data movement, a large number of tabular data have been published on the Web and Open Data Portals. Such tabular data contains valuable information and could be potentially useful in various fields, such as health, food security, climate change, resource management, smart cities and so on. Additionally, our society has become data-driven, where more and more data expected to grow in the near future from large volume, variety, and velocity, as a result, it is promising for establishing transparency, improving the quality of human life, and inspiring business opportunities.

Although these tabular data offers huge potential, these data are difficult to use due to fragmentation, heterogeneous schema, missing or incomplete metadata. Therefore, the usability of tabular data is an open question and should be exploited. There are several works have been made on improving the usability of tabular data such as establishing standard policies for data providers, or performing automatic reconstruct semantic meaning for tabular data. The first solution on standard policies takes a lot of time, and effort and difficult to scale, while the second solution is more promising to automation, and scale-up.

This thesis focuses on the automatic reconstruct semantic meaning for tabular data. The methodology is to assign the elements of tabular data into semantic concepts in knowledge bases. As a result, the meaning of tabular data could be interpreted or inferred by knowledge base concepts, therefore, it is easy to use in other downstream applications.

In this thesis, we firstly review the table data annotation for data interoperability including matching tasks, challenges, possible applications. Additionally, we identify potential limitations of tabular data annotation: 1) common text-based approaches are less effective in annotating numerical attributes; 2) entity lookup on one search

engine is imperfect on the general and multi-language text. Then, we introduce the novel solutions to address these limitations of tabular data annotation. We introduce distribution-based similarities (DBS), and a deep similarity metric (EmbNum+) for numerical attribute annotation to address the first limitations, and MTab which is a general framework for tabular data annotation for the limitations 1 and 2.

The following describes the details of these methods.

First, we present a lightweight solution on semantic annotation called DBS for numerical attributes. Existing approaches rely on the p value of a statistical hypothesis test as a metric to estimate the similarity between numerical attributes, and then assign unknown attribute by the labeled attributes. However, the p value-based metrics strongly depend on assumptions about distributions and data domain. In other words, they are unstable for general cases, when such knowledge is undefined. We present effective metrics called distribution-based similarities (DBS) to address the limitations of p value-based metrics.

Second, we present an effective and efficient method called EmbNum+ which is an end-to-end system to learn a similarity metric directly from numerical attributes. EmbNum+ was inspired by deep metric learning approaches with which both representations and a similarity metric are learned without making any assumption regarding data; hence, enabling EmbNum+ to be more generalized with a variety of data types and distributions. Evaluations on many datasets of various domains show that EmbNum+ consistently outperformed other approaches in terms of effectiveness and efficiency.

Third, we present a general framework for tabular annotation called MTab. MTab combines the voting algorithm and the probability models to tackle bottleneck problems of tabular data annotation. Additionally, we also adopt more signals from table elements and introduce a novel scoring function to estimate the uncertainly from ranking. This system got the first prize for entity annotations (**CEA**), type annotations (**CTA**), and relation annotations (**CPA**) at the Semantic Web Challenge on Tabular Data to Knowledge Graph Matching, the 18th International Semantic Web Conference 2019.

List of Publications

Journal Paper

Phuc Nguyen, Khai Nguyen, Ryutaro Ichise, and Hideaki Takeda. Embnum+: Effective, efficient, and robust semantic labeling for numerical values. New Generation Computing, Nov 2019. ISSN 1882-7055. doi: 10.1007/s00354-019-00076-w.

Conference Paper

Phuc Nguyen, Khai Nguyen, Ryutaro Ichise, and Hideaki Takeda. Embnum: Semantic labeling for numerical values with deep metric learning. In Semantic Technology - 8th Joint International Conference, JIST 2018, Awaji, Japan, November 26-28, 2018, Proceedings, pages 119–135, 2018. doi: 10.1007/978-3-030-04284-4_9.

Technical and Poster Papers

1. Phuc Nguyen, Natthawut Kertkeidkachorn, Ryutaro Ichise, and Hideaki Takeda. Mtab: Matching tabular data to knowledge graph using probability models. CoRR, abs/1910.00246, 2019, Semantic Web Challenge on Tabular data to Knowledge Graph Matching, ISWC 2019. (technical paper). (**first prize**).

2. Phuc Nguyen, Natthawut Kertkeidkachorn, Ryutaro Ichise, and Hideaki Takeda. "MTab: Matching Tabular Data to Knowledge Graph with Probability Models." in the Fourteenth International Workshop on Ontology Matching, 2019. (poster paper).

3. Phuc Nguyen and Hideaki Takeda. Semantic labeling for numerical values: Distribution-based similarities. In Proceedings of the 47 Special Interest Group for Semantic Web and Ontology, Ishigaki, Japan, number 12, March 2019. (technical paper).

4. Phuc Nguyen and Hideaki Takeda. Semantic labeling for quantitative data using Wikidata. In Proceedings of the 45 Special Interest Group for Semantic Web and Ontology, Tokyo, Japan, number 4, August 2018. (technical paper).

Contents

List of Figures

List of Tables

1 Introduction

In this chapter, we begin with the motivation of tabular data annotation in Section 1.1 and thesis objectives in Section 1.2. After that, we summarize our contributions as well as our published works in Section 1.3. Finally, the organization of the entire dissertation is described in Section 1.4

1.1 Motivation

Tabular data is semi-structured data, widely used for many aspects of human life such as reports, notes, books, media, software, and many other places. It is an effective and efficient way to store and represent data for human consumption since its compact representation reflects the logical relation between columns, rows, and cells. In the era of computers and the Internet, tabular data is the most popular structure for relational databases, Web tables, spreadsheets, and Open Data.

Nowadays, with the vision of Open Data [1], a large number of tabular data have been published on the Web and Open Data Portals. Such tabular data contains valuable information and could be potentially useful in various fields, such as health, food security, climate change, resource management, smart cities and so on. Additionally, our society has become data-driven, where more and more data expected to grow in the near future from large volume, variety, and velocity. As a result, it is promising for establishing transparency, improving the quality of human life, and inspiring business opportunities.

Although these tabular data offers huge potential, these data are difficult to use due to fragmentation, heterogeneous schema, missing or incomplete metadata.

- **Fragmentation:** Since tabular data is designed for human consumption, data usually present in small tables with limited space as the use in documents, reports, books, websites. The fragmentation of tabular data becomes an immerse issue for data interoperability where metadata missing or incomplete. If tabular data contains only a small number of rows and columns, it is difficult to find the correct correspondences, which can lead to insufficient matching results.

- **Heterogeneous schema:** Each data resource is independently constructed by different people with different backgrounds, purposes, and contexts. Therefore,

[1] Open Data Vision: https://opendatabarometer.org

the use of vocabulary and schema structures might differ across various data resources. For example, one table attribute uses "population" as the table header label and another table attribute uses a "number of people." Do these two attributes labels share the same or different meanings? This "semantic heterogeneity" may lead to the propagation of misinformation in the data integration process.

- **Missing or incomplete metadata:** Metadata is structured information that contains semantic annotation for tabular data. It plays a crucial role in data interoperability since it helps to find, access, inter-operate and reuse on such tabular data. Unfortunately, many tabular data do not contain metadata, or many of them are missing or not using the standard concepts.

Regarding those issues, the usability of tabular data is an open question and should be exploited.

There are several works have been made on improving the usability of tabular data such as establishing standard policies for data providers [3], or performing automatic reconstruct semantic meaning for tabular data. The first solution on standard policies takes a lot of time, and effort and difficult to scale, while the second solution is more promising to automation, and scale-up.

This thesis focuses on the second direction: automatic reconstruct semantic meaning for tabular data by matching table elements of tabular data into semantic concepts into standard knowledge bases. Readers might interested in the review paper of Farber et al. [4] on the most common knowledge bases as DBpedia, Freebase (discontinued since 2015), OpenCyc, Wikipedia, and Yago.

Figure 1.1 illustrate an example of tabular data annotation where table elements are automatically matched to the corresponding concepts in standard knowledge bases. As a result, the meaning of tabular data could be interpreted or inferred by knowledge base concepts, therefore, it is easy to use in other downstream applications.

Prior studies on semantic annotation for tabular data focus on the data model of relational table type (a table has a core attribute where cell values in this attribute could be matched into entities of knowledge bases, and the relations between the core attribute and other attributes could be matched into predicates or properties) where table attributes could be mapped into ontologies or existing schema of knowledge

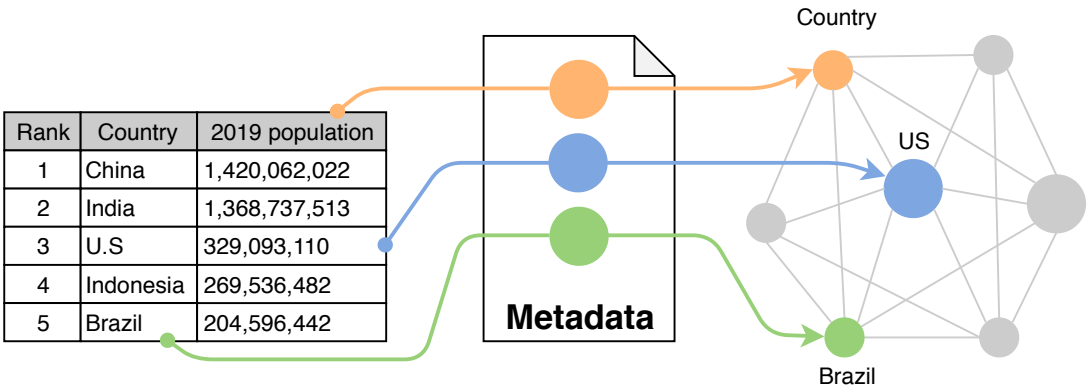


Figure 1.1: Illustration of tabular data annotation with knowledge graph matching

bases. These approaches first find the entity candidates from knowledge bases with table cell values, and then, rely on a relational data model to select the best entity candidates, and infer types, and relation candidates.

However, many tabular data cannot find candidates as the first step since table cell values are missing, ambiguous, or multi-languages, so that, entity candidates cannot be found directly using linguistic approaches. Moreover, many tables contain numerical attributes and numerical values which is difficult to find the relevant concepts in knowledge bases. According to an observation experiment on Open Data tables of Neumaier et al., 28% tables have missing headers, and many headers could be mapped using BabelNet[2] services (a multilingual lookup service) [5]. As a result, the problem of semantic annotation for such tables still an open question and need to be addressed.

1.2 Objectives

The objectives of this thesis firstly review the background of tabular data annotation for data interoperability including matching tasks, challenges, possible applications. Second, we identify potential limitations of tabular data annotation: 1) common text-based approaches are less effective in annotating numerical attributes; 2) entity lookup on one search engine is imperfect on general, multi-language text. Then, we aim to provide novel methods to improve semantic annotation for tabular data

[2]BabelNet Link: https://babelnet.org/

(MTab), in particular, the treatment of numerical attribute is the main focus (DBS, and EmbNum+).

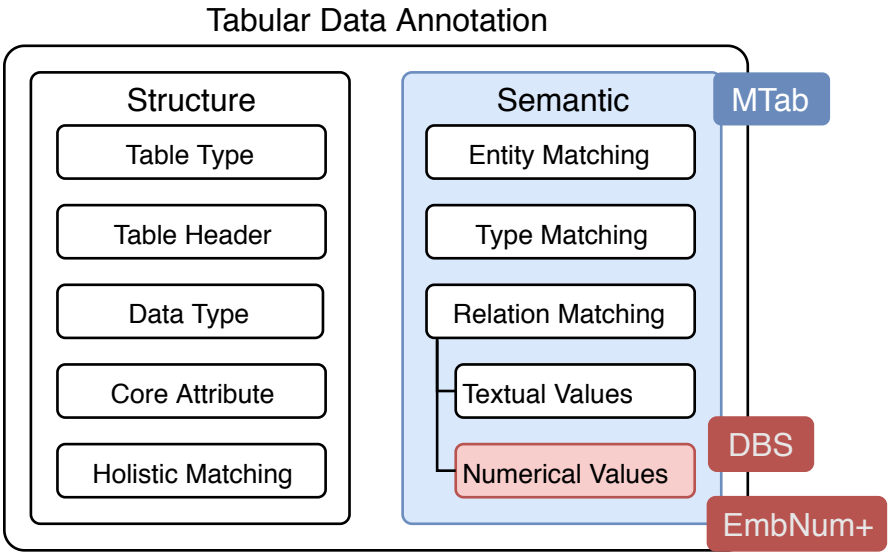


Figure 1.2: Objectives of the thesis

Figure 1.2 depicts the tabular data annotations tasks, and the main focus of this thesis. Regarding the first limitation, we introduce distribution-based similarities (DBS), and EmbNum+ for semantic labeling of numerical attributes. Then, we present the MTab system, which is a general framework for tabular data annotation which addresses limitations 1 and 2.

DBS is a lightweight solution on semantic annotation for numerical attributes. Existing approaches rely on the p value of a statistical hypothesis test as a metric to estimate the similarity between numerical attributes, and then assign unknown attribute by the labeled attributes. However, the p value-based metrics strongly depend on the assumptions about the distribution and data domain. In other words, they are unstable for general cases, when such knowledge is undefined. We present effective metrics called Distribution-based Similarities (DBS) to overcome the limitations of p value-based metrics.

EmbNum+ is an effective and efficient method designed for the task of semantic labeling for numerical attributes. EmbNum+ was inspired by deep metric learning approaches with which both representations and a similarity metric are learned

without making any assumption regarding data; hence, enabling EmbNum+ to be more generalized with a variety of data types and distributions. Evaluations on many datasets of various domains show that EmbNum+ consistently outperformed other approaches in terms of accuracy and efficiency.

Third, we present a general framework for tabular annotation called MTab. MTab combines the voting algorithm and the probability models to tackle bottleneck problems of tabular data annotation. Additionally, we also adopt more signals from table elements and introduce a novel scoring function to estimate the uncertainly from ranking.

1.3 Contributions and Published Works

This thesis provides the following contributions:

1. We introduce DBS [6] [3] which is a lightweight solution on semantic annotation for numerical attributes. DBS tackles the problems of p value-based metrics for the task of semantic labeling for numerical attributes.

2. We introduce EmbNum+ [7] which is an end-to-end deep metric learn directly from numerical attributes. As our experiments show that EmbNum+ outperformed other state-of-the-art systems in terms of effectiveness, efficiency, and robustness in all tested datasets.

3. We provide three new numerical attributes datasets which enables rigorous evaluation for numerical attribute matching: two synthesis datasets, e.g., DBpedia numerical knowledge bases (NKB), and Wikidata NKB, and one real-world dataset, e.g., Open Data (extracted from five Open Data Portals) [4]. These datasets cover a wide range of challenges then they enabled a more rigorous analysis on the task of semantic annotation for numerical attributes.

4. We introduce MTab [8] which is a general framework for tabular data matching to knowledge bases. MTab is a novel method tackle two bottleneck problems of tabular data annotation. Additionally, we also adopt more signals from table

[3] DBS demo: https://github.com/phucty/dbs
[4] Datasets: https://github.com/phucty/embnum

elements and introduce a novel scoring function to estimate the uncertainly from the ranking of lookup results.

1.4 Thesis Outline

This dissertation is organized as follows.

- **Chapter 1** describe the thesis motivation of tabular data annotation and the objectives so that we address in this thesis. We also provide a summary of the thesis contributions and our published works.

- **Chapter 2** presents an overview of the problem of tabular data annotation including motivation, and possible downstream applications, use-cases. This section also provides the formal definition as well as the notation about the annotation tasks. We also present an overview of the state-of-the-art techniques, and we also discuss their limitations in this section.

- In **Chapter 3**, we present a lightweight solution on semantic annotation for numerical attributes. Existing approaches rely on the p value of a statistical hypothesis test as a metric to estimate the similarity between numerical attributes, and then assign unknown attribute by the labeled attributes. However, the p value-based metrics strongly depend on the assumptions about the distribution and data domain. In other words, they are unstable for general cases, when such knowledge is undefined. We present effective metrics called Distribution-based Similarities (DBS) to overcome the limitations of p value-based metrics.

- In **Chapter 4**, we present an effective and efficient method called EmbNum+ which is an end-to-end system to learn a similarity metric directly from data. EmbNum+ was inspired by deep metric learning approaches with which both representations and a similarity metric are learned without making any assumption regarding data; hence, enabling EmbNum+ to be more generalized with a variety of data types and distributions. Evaluations on many datasets of various domains show that EmbNum+ consistently outperformed other approaches in terms of accuracy and efficiency.

- In **Chapter 5**, we present a general framework for tabular annotation called MTab. MTab combines the voting algorithm and the probability models to tackle bottleneck problems of tabular data annotation. Additionally, we also adopt more signals from table elements and introduce a novel scoring function to estimate the uncertainly from ranking.
- **Chapter 6** concludes the thesis and discuss some direction of our future work.

2

Tabular Data Annotation Background

This chapter is organized as follows. Section 2.1 describes the two potential problems tabular data annotation. The following Section 2.2 provide some definitions of fundamental concepts of tabular data annotation. Section 2.3 summary the previous studies on the annotations tasks. In Section 2.4, we provide potential application for tabular data. Finally, we discuss about the limitations and and introduce the focus tasks in this thesis in Section 2.5.

2.1 Background

Over the last decade, we have seen tabular data being populated more and more on the Internet such as Web Tables, or Open Data Portals. Such tabular data usually do not have a machine-understandable capability, it means that the data do not come with fully annotated metadata. Tabular data could be indexed, and retrieved with keyword-based techniques, but it does not allow for a higher-level understanding of context or semantics of data. To unlock this potential, it is important to match elements of tabular data into knowledge bases, which contains a predefined knowledge about a specific domain or general knowledge of the world. It enables other downstream applications to access the content of tabular data without understanding the structure or context.

The current approaches on tabular data annotation focus on relational tables, where table cells could be matched into entities in knowledge bases. However, there are a lot of tabular data which do not contain textual content (or contains but can not be matched into knowledge graphs), but they have a large number of numerical values. Many methods ignore those tabular data, as a result, it leads to mispropagation in the annotation process. In this thesis, we proposed two methods to address the semantic annotation for numerical values in tables. It could be used in difficult tables where there are no matching entities but contains numerical values (DBS [6], EmbNum+[7]) or it also provides useful information to enhance the annotation performances in general tabular data (MTab [8]).

Moreover, the representation of table cells is complicated, since it could be in multiple-language, with strange encoding. Previous approaches performed lookup those table cells directly on knowledge bases (Elastic Search, DBpedia lookup or, Spotlight) then lead to no retrieval results. In this thesis, we proposed the MTab

framework for solving this problem. We perform language detection on table cells and lookup with language parameters on multiple-lookup services which yield really promising performances in entity lookup.

2.2 Definitions

In this section, we define fundamental concepts will be used in the entire of this thesis.

Definition 2.1 *(Tabular data). Tabular data is data structured into tables.*

Definition 2.2 *(Table). A table (denoted as T) is a two-dimensional tabular structure consisting of an ordered set of N rows and M columns.*

Definition 2.3 *(Row). A row (denoted as n_j) is a row of table where $i = 1...N$).*

Definition 2.4 *(Column). A column (denoted as m_j) is a column of table where $j = 1...M$.*

Definition 2.5 *(Cell). A cell (denoted as $T_{i,j}$ is an intersection between a row and column, with its values (denoted as $c_{i,j}$) as a number, string or an empty value.*

According to Nishida et al., there are six table types, i.e., vertical and horizontal relational tables, vertical and horizontal entity tables, matrix tables, and other tables [1]. The following definitions are on the table types.

Definition 2.6 *(Genuine tables). A genuine table is a table which contains semantic triples of knowledge graph in a from of <subject, predicate, object>*

Definition 2.7 *(Relational table). A relational table is a genuine table which contains a core attribute where other attributes are other predicate information of this core attribute.*

Definition 2.8 *(Vertical relational table). A vertical relational table is a relational table that the core attribute present in one column.*

An example of a vertical relational table is shown in Figure 2.1.

Definition 2.9 *(Horizontal relational table). A horizontal relational table is a relational table that the core attribute present in one row.*

name	place	death
A. Prior	New Zealand	1969
A. Drews	Uetersen	1935

Figure 2.1: Example of a vertical relational table

name	A. Prior	A. Drews
place	New Zealand	Uetersen
death	1969	1935

Figure 2.2: Example of a horizontal relational table

An example of a horizontal relational table is shown in Figure 2.2.

Definition 2.10 *(Entity table). A entity table is a table where describe properties for single entity. It also have the vertical entity table, and horizontal entity table.*

Examples of entity tables are shown in Figure 2.3a (vertical entity table) and Figure 2.3b (horizontal entity table).

place	New Zealand
death	1969

(a) Vertical Entity Table

place	death
New Zealand	1969

(b) Horizontal Entity Table

Figure 2.3: Examples entity tables: Vertical 2.3a, and Horizontal 2.3b

Definition 2.11 *(Matrix table). A matrix table does not contain information about context, or name of the attributes. It is mostly used to provide statistics.*

An example of a matrix table is shown in Figure 2.4.

The following definitions are on the elements inside tables.

Definition 2.12 *(Table attribute). A table attribute is a column of a vertical relation table, row of a horizontal relation table or a row (column) of a matrix table.*

country	2019	2015	2010
Japan	126	127	128
US	330	321	310

Figure 2.4: Example of a matrix table

name	place	death
A. Prior	New Zealand	1969
A. Drews	Uetersen	1935

name	A. Prior	A. Drews
place	New Zealand	Uetersen
death	1969	1935

country	2019	2015	2010
Japan	126	127	128
US	330	321	310

Figure 2.5: Examples of table attributes

Examples of table attributes are shown in Figure 2.5.

Definition 2.13 *(Textual attribute). A textual attribute is a table attribute where all values of this attribute are textual.*

Definition 2.14 *(Entity attribute). An entity attribute is a table attribute where the cell values of this attribute could be matched into entities in knowledge graphs.*

Definition 2.15 *(Numerical attribute). A numerical attribute is a table attribute where all values of this attribute are numerical.*

2.3 Annotation Tasks

This section introduces the tasks and previous methods of tabular data annotation. These include the annotation tasks on table structure and table semantic as Figure 2.6. The annotation of table structure include classification of tabular data into different table types (Section 2.3.1), recognition of table headings (Section 2.3.2) or data type of table attributes, table cells (Section 2.3.3), and detection of a subject column or the core attribute where all cells could be matched into entities of knowledge bases 2.3.4. In

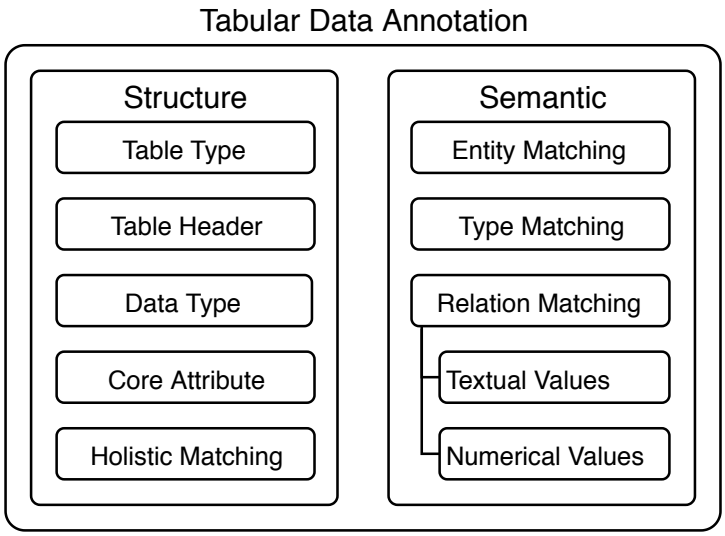


Figure 2.6: Tabular data annotation tasks

Section 2.3.5, holistic matching is described. The semantic annotation tasks where table elements are matched into knowledge bases will be introduced as entity matching (Section 2.3.6), type matching (Section 2.3.7), and relation matching (Section 2.3.8).

2.3.1 Table Type Classification

In this section, we introduce the task of table type classification for tabular data. The goal of this task is to assign a table type to a given tabular data. These types allow us to distinguish between whereas tables used for layout purposes or containing data, horizontal or vertical, entity or relational. There are many studies on understanding table type taxonomy, in this thesis, we adopt the table types taxonomy mentioned in Nishida et al. [1].

Figure 2.7 depicts the hierarchy of table types. In genuine type, the table contains a semantic form of <subject, predicate, object>. The relational type have a table core attributes, where relational vertical is that the core attribute present in one columns (Figure 2.1), and relational horizontal is that the core attribute present in one row (Figure 2.2). The entity type is a table type where describe the attribute for single entity. Figure 2.3b, and Figure 2.3b are the examples of entity vertical, and entity horizontal tables. The matrix types usually do not contain the information about

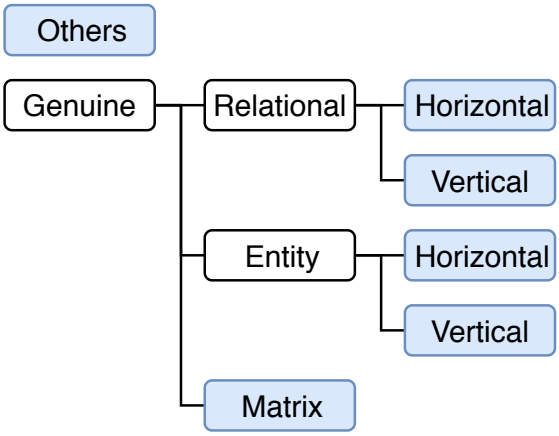


Figure 2.7: Table type hierarchy (Adapted from Nishidata et al. [1])

context, or the names of attributes (predicates) (Figure 2.4). Such matrix tables are used to present the statistics. The other types are those table types used for layout purposes, or other genuine types [9] mentioned *enumeration*, *calendar*, and *form*.

Several approaches used heuristics to classify table types such as if the size of tables is too small, tables contain other tables in their cells or the value in one cell are too long, they are considered as other types [9, 10, 11]. To recognized the other types, the supervised classification is used to classify the table types with handcrafted features [9, 10, 11] or neural networks [1].

There are many novel supervised methods, however, such training datasets usually do not make published. Therefore, the comparability and benchmark of these studies are limited.

2.3.2 Table Headings Detection

Table headings contain the names of table attributes. These names contain schema information, and they need to be annotated differently from the other content of tables (data). Usually, the problem of table headings detection is performed before doing data matching. Due to the heterogeneous table schema, table headers could have complicated structures. Headers could available or not available. Headers could have a hierarchy structure, as a result, it could present in one row or multiple rows. Figure 2.8 depicts an example of table headings where they are located at the beginning of the

table and having a hierarchy structure.

地域 Area			人口 Population		平成17年～22年の人口増減（－は減少） Population change, 2005-2010 (- decrease)		人口の都道府県別割合(%) Percentage of total population	人口密度（1km²当たり） Population density (per km²) 1)	世帯数 Households	
			平成22年 2010	平成17年（組替） (readjusted)	実数 Number	率(%) Rate			平成22年 2010	平成17年（組替） (readjusted)
全国		Japan	128,056,026	127,767,994	288,032	0.2	100.0	343	51,951,513	49,566,305
市部		All shi	116,153,998	115,503,691	650,307	0.6	90.7	537	47,672,722	45,360,720
郡部		All gun	11,902,028	12,264,303	−362,275	−3.0	9.3	76	4,278,791	4,205,585
01	北海道	Hokkaido	5,507,456	5,627,737	−120,281	−2.1	4.3	70	2,424,073	2,380,251
02	青森県	Aomori-ken	1,373,164	1,436,657	−63,493	−4.4	1.1	142	513,311	510,779

Figure 2.8: Example of table headings

Most studies used heuristics to detect table headings. The simplest heuristic is that headers are located in the first row of table [10, 12, 13]. Another heuristic is that the header rows have a different format with other parts of tables, therefore, we can use background color, or font formatting to recognize headers [14, 15, 16, 17]. Changing the data type from column cells also is a good signal to recognize table headers.

Yoshida et al. used a set of term examples in tables to distinguish header and non-header content [18]. Wang et al. use Probase [1] as external knowledge bases to query table headers [17].

In summary, the most common approaches use the position or table format, cell data type, or external knowledge to recognize the table headers.

2.3.3 Table Data Type Detection

Data type detection for table cells is an important task in tabular data annotation since the results might help to improve the performances of other tasks. Table cell values are encoded as strings, therefore it has some problem of understanding non-textual values which can be presented as various formats. Date presentation, for example, is one of this problem, it could be presented as "16 November 2019" or "2019/11/16".

There are several studies on data type detection for tabular data. Mulwad et al. use regular expressions to distinguish objects or literals [19]. Ritze et al. distinguish three different types such as date, string and numeric using a cascading strategy (parse

[1]Probase link: https://www.microsoft.com/en-us/research/project/probase/

each value in one specific type, and continues other types if it fails) [13]. Kim and Lee proposed 15 different data types such as image, form, time, date, month, day, string, number blank, temperature, voltage, weight, currency, percentage, and postal code [20]. The authors used predefined textual patterns and keywords to recognize the data types. Zhang [21] used regular expressions to determine data types such as empty, named entity, number, date, long text, and others. The final data type for table attributes is voted by majority data types of all cell values in these attributes.

Duckling Tool (Wit.ai) [2] is an open-source project for data type parser. Duckling used a probabilistic context-free grammar as their main technique for data parsing, and it can handle multi-languages. The 13 data types that ducking can handle are time, temperature, number, ordinal, distance, volume, amount-of-money, duration, email, URL, and phone number.

In summary, data type detection is important for a correct interpretation of table values. In the MTab system, we used the Duckling Tool to predict data types for table cells, and also using majority voting to re-estimate the final data type for table attributes which is similar to Zhang approach [21].

2.3.4 Core Attribute Detection

In this section, we describe the core attribute detection which is a task of find subject column (vertical type) or subject row (horizontal type) in the relational tables. However, most studies focus on the subject column in the vertical relational tables as the core attribute. A simple heuristic is used so that the left-most column is the core attribute [10, 12].

Venetic et al. extended this heuristic so that the column data types are not numeric or date, the number of unique values in column [13, 22]. Wang et al. [17] perform entity linking which lookup all cells in the same columns to get corresponding entities from a knowledge base. The largest matched entities are the core attribute. Zhang [21] used hand-draft features of table columns to recognize the core attributes such as the fraction of empty or unique cell values, the index of column (left-most), acronyms, IDs available or not, the overlapping rate between column headers and surrounding text.

In summary, common approaches in the core attribute detection used column

[2] Duckling Tool Link: https://github.com/facebook/duckling

position, uniqueness, or matched entities to predict whereas a column is the core attribute or not. The results of this task are really helpful for other tasks especially for the task of knowledge base population. The core attribute assumes that all cell values of a core attribute could be matched into entities in knowledge bases, if some cells cannot be matched, they are new entities and could be added into the knowledge bases.

2.3.5 Holistic Matching

Holistic matching is a task of merging tabular data with the assumption that there is a shared schema between tables. This task is especially helpful for those fragmented tables where a big table is separated into many small tables for the presentation purpose (limited space, for human consumption, papers, documents, or websites).

Ling et al. first proposed the problem of unioning tables with identical schemas [23]. The authors augment the union table with new attributes to guarantee the semantic consistent. Lehmberg and Bizer extended Ling et al. work using schema-based and instance-based matching techniques [24].

2.3.6 Entity Annotation

In this section, we describe the entity annotation task which is a task of matching table cells into corresponding entities in a knowledge base.

Previous approaches on entity annotation firstly perform the lexical similarity on the table cell value or other cell values in the surrounding context to find entity candidates. Sekhavat et al. used exact lexical matching to find entity candidates [25], while other approaches used other string similarity metrics [13, 16, 19, 26, 27, 28, 29].

Zhang considers table cell value, and other neighbor cells (same row or column) to estimate contextual similarity [21]. There were several studies lookup different sources such as Wikipedia to find entity candidates [19, 26], Mulwad et al. also considers the popularity of Wikipedia pages of entities such as PageRank. Recently, there has been an increasing interest in using entity embedding to find entity candidates [29], however, the performances of embedding approaches are still limited in comparison with lookup or similarity matching.

As tabular data is a semi-structured data where table elements have shared relations of <subject, predicate, object>, it is helpful to use the results from other tasks such as

type annotation or relation annotation to improve the result of entity annotation. Common approaches model the interaction between the three tasks of entities, types, and relations annotation [13, 16, 19]. The annotation of entities in the same column usually has the same type annotations or relation annotations. Another approaches based on iterative matching where the sub-sampling (rows) of tables is first matched in turn then updated each iteration until the type annotations are converged [21].

2.3.7 Type Annotation

Type annotation is a task of matching table attributes into classes in knowledge bases. The result could be a single value (the exacted matched class) or multiple values (the exacted matched classes and these ascendance classes). In the relational table, finding a type for the core attribute could be considered as the type for the whole table.

Most studies in type annotation used the similarity metric between table headers and knowledge base classes [16, 30] or use directly the knowledge base lookup service as [17]. Other approaches used column values to find the corresponding types in knowledge bases by collective similarity measures [13, 28, 30, 31]. In particular, the metric as TF-IDF or cosine similarity could be used on all concatenated values of the table columns and knowledge base classes. Fan et al. proposed a hybrid machine-crowdsourcing system where a machine learning approach is used in easy cases and difficult cases are annotated by a crowdsourcing service[31].

Several studies used the entity annotation as the input for the problem of type annotation. Zhang used the union of all entities classes as the type annotation [21], majority voting on entity class in a column is used [13, 19, 27], or measure the hierarchy distance between entity classes and candidate classes [16]. Entity embedding also used to predict table attribute types [32, 33].

2.3.8 Relation Annotation

In this section, we describe the task of relation annotation using textual values and numerical values.

Regarding textual information, the relations between the core attribute and other attributes are matched into predicates (properties) in knowledge bases. In data-based approaches, each pair of core attribute and cell values is compared with the pairs of a

subject, object in knowledge bases triples (subject, predicate, object>, then the relation is aggregated from matched predicate using majority voting [21]. The other is that table headers are used to find the corresponding property labels in knowledge bases using string similarity metrics [21, 30]. The relation annotation result could be inferred from the result of type annotation with "Domain" and "Range" relation of property in knowledge bases [30], or majority voting of properties by entity pairs of two columns [16, 19, 21, 28, 34].

The numerical information is quite different from textual information because of numerical value representation. It is difficult to find the corresponding meanings since 1) the numerical representation of knowledge base value could be represented in a different ways (such as scaling meter or centimeter), different values (such as population of this year, and next year), or 2) the different meaning numerical values in knowledge bases are very similar with the query numerical values (such as population and areaLand).

Most of the studies on semantic annotations for numerical attributes consider these numerical values as a collection or a bag of numbers and search it on numerical knowledge bases (extracted from knowledge bases) in terms of similarity metrics. The authors used the p value of statistical hypothesis tests to estimate the similarity between numerical values [2, 5, 35, 36]. Neumaier et al. created a numeric background knowledge base from DBpedia [5] with hierarchy clustering approach, the background knowledge base allows inferring detail contexts of numerical attributes such as city temperate, building height, the height of a human, and so on.

Overall, these studies in relation annotation rely on an assumption that table attributes could be matched into predefined types or properties of knowledge graphs. However, the real-world knowledge graphs have a problem of incompleteness, and it is not easy to match if the table attributes are not available in knowledge graphs.

2.4 Applications of Tabular Data Annotation

In this section, we present some possible applications of tabular data annotation. Most use cases of tabular data annotation are on information retrieval and knowledge management.

Annotation data could be used as a fact searching [37] or to enhance the current

searching engines enable them to be retrieved table data. Google Fusion Table [3] is a public service allows users to find, merge, visualize, and share tabular data [27]. Another application is on knowledge exploration as Chirigati et al. [38], where the authors proposed a method to explore IS-A and HAS-A relationships from Web tables.

Tabular data could be used in the problem of table extension when we want to extend a table with additional rows, or columns. InforGather relies on a collection of related tables on the Web to populate tables in terms of entity attribute names or attribute values of entities [39]. Bhagavatula et al. search for matching tables in WikiTables and then perform join techniques to extend the input table [26]. Lehmberg et al. introduce the Mannheim Search Join Engine which focuses on large scale heterogeneous tabular data sources [40].

Tabular data could be used to construct or extend knowledge bases such as DBpedia, Wikidata, Freebase, YAGO and so on. Many studies proposed annotation methods for semantic interpretation which are matching table elements into knowledge bases, and complete missing values in knowledge bases [17, 21, 25, 41, 42, 43]. In TableNet, the Wikipedia tables could be interlinked with IS-A and HAS-A relations to construct a knowledge graph [44].

Finally, such tabular data could be used for question answering system, for example, answering factual user questions [37, 45], or quantitative questions [46].

2.5 Current Limitations of Tabular Data Annotation

Most approaches rely on textual information which could be found in tabular data. However, tabular data contains not only in English, but it could also be represented in other languages, which currently approaches just focus on processing English tabular data. For example, DBpedia lookup which is one of the most popular lookup services for finding DBpedia concepts is indexed in English. As a result, when we look up tables in a different language, we can not get the answers. In this thesis, the problem of entity lookup is addressed by firstly, recognizing the languages used in tabular data, then, lookup in many services with language parameters to find corresponding entities. We will introduce it in MTab which yields promising performances in entity lookup

[3] Google Fusion Table Link: https://fusiontables.google.com/

(Section 5).

Another problem is that table values do not usually have rich textual descriptions or cannot be matched into knowledge bases using linguistic approaches, specifically where tables contain many numerical attributes. Those numerical attributes are missing or ambiguous headers such as ID, code, abbreviation. According to mapping on 1200 tables of Neumaier et al. [5], only 20% header labels could be mapped into knowledge bases. Therefore, the problem of annotation for numerical attributes is an open problem, and need to be addressed.

Most studies on semantic annotations for numerical attributes used the p value of hypothesis test as a similarity metric however, such metric are strongly depend on assumption about data type, and data distribution. As a result, it is not robust in general data, or unknown data where prior data knowledge is unknown. In this thesis, we proposed two methods to address the semantic annotation for numerical values in tables. On the one hand, it could be used in difficult tables where there are no matching entities but contains numerical values (DBS [6], EmbNum+[7]). On the other hand, it also could be used to enhance the annotation performances in general frameworks (MTab [8]).

3

Distribution-based Similarities for Numerical Attribute Annotation

In this section, we first present the background of semantic labeling for numerical values in Section 3.1, and related work in Section 3.2. Section 3.3 introduces the novel distribution based methods derived from the norms of the inverse transform sampling of attribute distributions to estimate the similarity between numerical attributes. Section 3.4 describes the experiments of DBS in comparison with other p value-based approaches. Finally, we conclude with Section 3.5.

3.1 Introduction

In recent years, there has been an increasing interest in numerical semantic labeling for tabular data where numerical values from table columns are matched to the semantic labels in knowledge bases. It enable integrated numerical data and hence could be potentially useful for other applications such as table search [47, 48], table extension [49], completion[50], or knowledge base construction as used in DBpedia [51], YAGO [25], and Freebase [52].

A common work-flow is the retrieval setting in which the label of a query column is assigned by that of the most relevant columns in labeled data with respect to a specific similarity or distance metric. However, how to select a good similarity or distance metric for numerical attributes is a difficult challenge because of several reasons.

Issue 3.1 *The numerical values of attributes rarely have the same set of values as the relevant values in knowledge bases.*

Issue 3.2 *The size of attributes could vary from a few to millions of numbers. It is hard to use directly apply the normalized vector spaces as similarity metrics.*

Issue 3.3 *In general cases, we do not have the predefined knowledge about distribution and type of data.*

Previous approaches used the p value of a statistical hypothesis test as a metric to measure the similarity between numerical attributes [2, 5, 36]. The p value-based similarity address the first (Issue 3.1) and second issue (Issue 3.2), however it cannot be used in the third issue (Issue 3.3). In fact, a statistical hypothesis test strongly depends on assumptions regarding the distribution and type of data. For instance,

these data attributes have to be drawn from a specific form of distribution (e.g., normal distribution or uniform distribution) or data types (e.g., continuous or discrete). However, determining the form of distributions and data types of unknown numerical attributes is a difficult challenge. As a result, a proper hypothesis test cannot be easily selected when we do not have such predefined information.

Moreover, there is controversy when using p value as a similarity [6, 53, 54, 55]. Other baselines define the null hypothesis H_0 is that the two numerical attributes are similar in terms of semantic meaning. The p values from hypothesis tests are interpreted as metrics to measure the level of similarity. In other words, the p-value measures the probability of H_0 is correct. However, this use of p value is revealed as misuse by many statisticians [54, 55]. Overall, it is open to doubt whether the p value-based metrics are suitable for the problem of semantic labeling for numerical attributes.

To address these limitations, we propose distribution-based similarities (DBS) which is a method to estimate the similarity between numerical attributes without using p-value based metrics. In particular, DBS metrics are calculated from the empirical cumulative distribution of numerical attributes without making any assumption about data type and data distribution.

We evaluate the performance of DBS against two baseline approaches SemanticTyper [2], and DSL [36] on City Data and Open Data. The overall results show that DBS outperforms the baselines in a large margin. A demo of semantic annotation for numerical attributes between DBS and other baselines are available at this link[1].

3.2 Related Work

In this section, we present the previous approaches for semantic labeling with textual information (Section 3.2.1) and numerical information (Section 3.2.2).

3.2.1 Semantic Labeling with Textual Information

Several attempts have been made to assign semantic labels to for table attributes using the information on header labels and textual values [13, 51, 56]. The most common

[1]DBS demo: https://github.com/phucty/dbs

approaches use entity linkage for mapping textual values of attributes to entities in a knowledge base. The schema of entities is then used to find the semantic concept for table attributes. Additional textual descriptions of tables have also been considered to improve the performance of the labeling task [17, 22, 57]. However, it is not easy to apply these approaches are to numerical attributes because the unknown attribute rarely has the same set values with knowledge-base values [2]. To build an effective integrated system, taking into account semantic labeling using numerical information is necessary.

3.2.2 Semantic Labeling with Numerical Information

Most researchers on semantic labeling with numerical information use statistical hypothesis tests as similarity metrics to compare the similarity of numerical attributes. Stonebraker et al. proposed a method for schema matching using decisions from four *experts* [35]. One decision used the *t-test* statistic of the Welch' s t-test [58] to measure the probability that two numerical attributes were drawn from the same distribution. The t-test value is calculated based on the means and variances of two numerical attributes, which have to follow a normal distribution. However, numerical attributes do not always follow a normality assumption; therefore, applying a t-test as a similarity metric is not appropriate for the non-normality numerical attributes.

To address the limitations of the t-test, Ramnandan et al. proposed SemanticTyper to find a better statistical hypothesis test [2]. They tested on the Welch' s t-test [58], the Mann-Whitney U test [59], and the Kolmogorov Smirnov test (KS test) [58]. The authors reported that using the p value of the KS test archived the highest performance in their experiments than the two other tests. However, the Mann-Whitney U test and the KS test are used assuming that the two tested numerical attributes are continuous. Therefore, these tests are often not very informative when the distribution of tested attributes are assumed to be discrete.

Minh Pham et al. extended SemanticTyper (called DSL) by proposing a new similarity metric that is a combination of the KS test and two other metrics: the Mann-Whitney test (MW test) and the numeric Jaccard similarity [36]. Their experiments showed that using this combined metric provided better results over using only a KS test. However, performing multiple hypothesis tests is computationally intensive and it

is not easy to learn a robust metric when we only have a small number of training data.

Neumaier et al. created a numeric background knowledge base from DBpedia [5]. Given an unknown numerical attribute, they also used the p value of the KS test as a similarity metric to compare with each labeled attribute in the numeric background knowledge base. In fact, the size of attributes varies from a few numerical values to a few hundred thousand numerical values. Therefore, keeping the original numerical values of attributes in background knowledge bases is memory intensive. Moreover, it is very time-consuming to conduct the KS test between the unknown numerical attributes with all attributes in the background knowledge base since the computational overheads of the KS test are strongly based on the size of compared attributes.

Overall, the similarity metrics used with these approaches are hypothesis tests, which are conducted under a specific assumption regarding data type, and data distribution. In contrast to these approaches, we propose the new distribution-based similarities calculated from the empirical cumulative distribution of numerical attributes without making any assumption about data type and data distribution.

3.3 Distribution-based Similarities Approach

In this section, we introduce the new categories of similarity metrics, called Distribution-Based Similarities (DBS). The similarities are derived from a norm of the inverse transform sampling of numerical attributes.

The overall framework of semantic labeling with DBS shows in Figure 3.1. The framework consists of two phases. The first phase involves data preparation and knowledge base construction while the second phase is semantic labeling.

In the first phase, given labeled numerical attributes, the attribute transformation converts these labeled attributes from numerical values into distribution presentations (Section 3.3.1). Then, these distribution presentations are stored in the knowledge base for future similarity comparison.

In the second phase, the numerical values of an unknown attribute are standardized with attribute transformation. Then the similarity search module is used to calculate the similarities between these distribution representations. In this section, we consider three typical distance of the Minkowski distance: the Manhattan distance (called DBS_1),

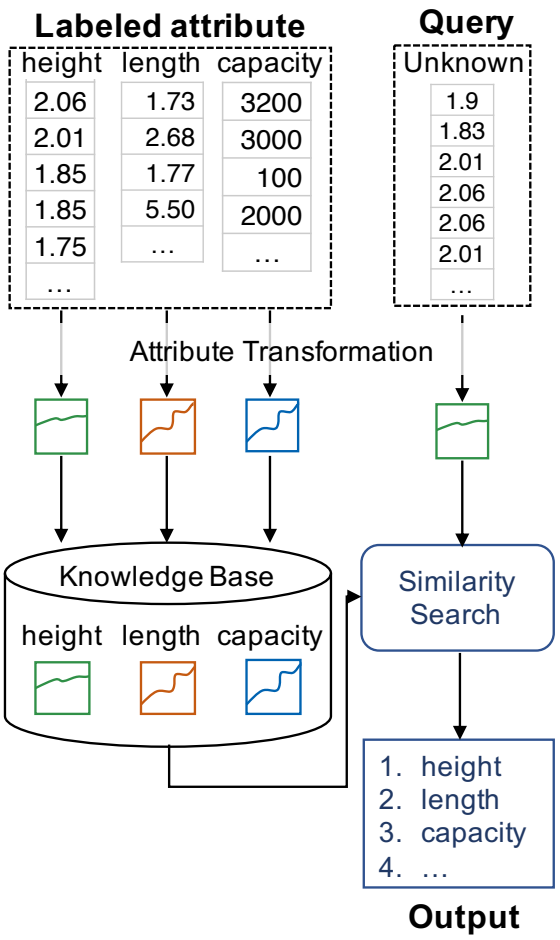


Figure 3.1: Semantic labeling framework for numerical attributes with DBS

the Euclidean distance (called DBS_2), and the Chebyshev distance (called DBS_∞). After the similarity searching process, we have a ranking list of semantic labels ordered by their corresponding similarity scores.

Figure 3.2 illustrate the DBS metrics. Intuitively, DBS metrics are average distances between the sampling CDF of the two attributes attribute. Overall, DBS address all the three mentioned issues: Issue 3.3, 3.1, 3.2.

1. **Issue 3.1:** The similarity is derived from distributions of numerical attributes, therefore it is not necessary the assumption that the values of numerical attributes have the same set of values.

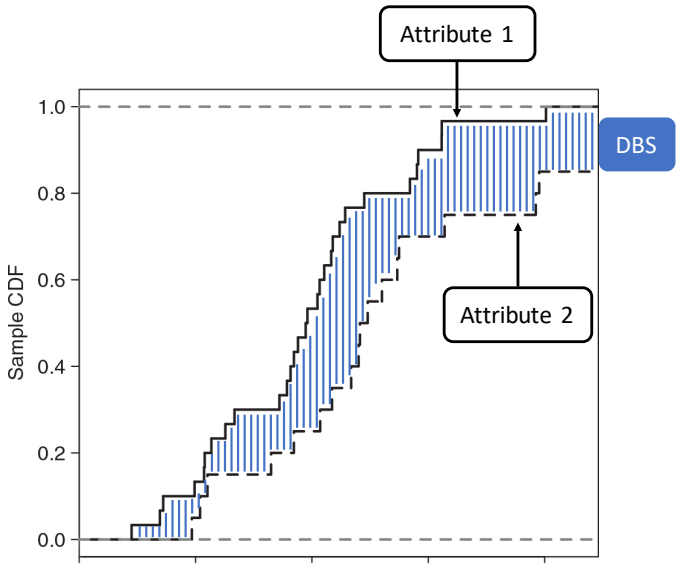


Figure 3.2: Illustration of DBS metrics

2. **Issue 3.2:** In DBS, we introduce an attribute transformation (Section 3.3.1) to transform the list of numerical values to a distribution representation as well as standardize the input size of numerical attributes. Therefore, after transformation, the numerical attribute has a representation as a vector with h size.

3. **Issue 3.3:** DBS is derived from the empirical distribution of numerical attributes without the need to make any assumption regarding data type or data distribution.

3.3.1 Attribute Transformation

In this section, we describe the transformation of numerical attributes to standardize the input size, for the representation learning. Attribute transformation is an important module because the representation learning requires a standardized input size, and the size of numerical attributes could vary from a few to thousands of values.

We use inverse transform sampling [60] (Section 3.3.1) to standardize the input size and transform numerical values into forms of distribution presentations. This technique is chosen because it retains the original distribution of a given list of numerical values. In Section 3.3.1, we empirically showed that the output from the inverse transform sampling is better than the usual random-choice sampling technique.

After transformation, the list of numerical values is sorted in a specific order to leverage the capability of the CNN network to learn representations from distribution presentations.

Given an attribute a having numerical values $V_a = [v_1, v_2, v_3, ..., v_n]$, the objective of attribute transformation is the $trans(V_a)$ function, which is defined as follows.

$$x = trans(V_a) = x_{icdf} \tag{3.1}$$

The transformation function $trans(\cdot)$ converts V_a into x, where $x \in \mathbb{R}^h$. The list of values $x_{icdf} \in \mathbb{R}^h$ is obtained by the transformation using the inverse transform sampling (Section 3.3.1) on numerical values. The inverse transform sampling is described as follows.

Inverse Transform Sampling

Let a be an attribute with numerical values $V_a = [v_1, v_2, v_3, ..., v_n]$. We treat V_a as a discrete distribution so that the CDF of $v \in V_a$ is $cdf_{V_a}(v)$ and expressed as follows.

$$cdf_{V_a}(v) = P(V_a \leq v), v \in V_a, cdf_{V_a} : \mathbb{R} \rightarrow [0, 1] \tag{3.2}$$

where $P(V_a \leq v)$ represents the probability of values in V_a less than or equal to v. The inverse function of $cdf_{V_a}(\cdot)$ takes the probability p as input and returns $v \in V_a$ as follows.

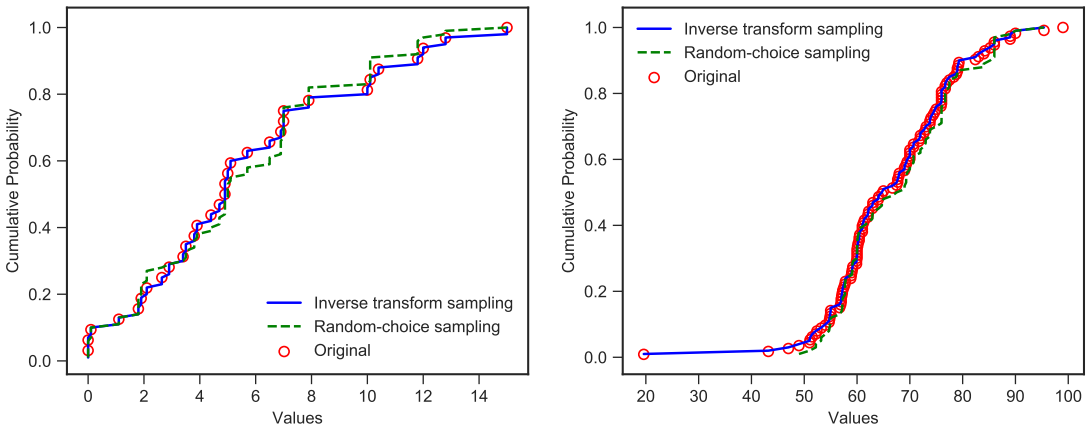
$$icdf_{V_a}(p) = cdf^{-1}_{V_a}(p) = min\{v : cdf_{V_a}(v) \geqslant p\}, p \in [0, 1] \tag{3.3}$$

We select h numbers (Section 4.4.3) from V_a where each number is the output of the inverse distribution function $icdf_{V_a}(p)$ with probability $p \in \mathcal{P} = \{\frac{i}{h} | i \in \{1, 2, 3, ..., h\}\}$. For example, when the input size $h = 100$, then we have $\mathcal{P} = \{0.01, 0.02, 0.03, ..., 1\}$. For each attribute $a \in A$, we have a list of values $x_{icdf} = \{v_1, v_2, v_3, ..., v_h\}$ that correspond to the given list of probabilities $\mathcal{P}$.

Transformation Analysis

To understand how well the samples of the inverse transform sampling fit the original data, we analyzed two attributes in City Data using the inverse transform sampling and

the random-choice sampling technique, which generates a uniform random sample from a given list of numerical values.



(a) decRainDays (b) aprHighF

Figure 3.3: Analysis of inverse CDF and random-choice sampling technique on the *decRainDays* (3.3a) and the *aprHighF* (3.3b) properties of City Data.

Figure 3.3 depicts the transformation results of two techniques on the *decRainDays* and the *aprHighF* properties of City Data. The distribution using the inverse transform sampling (blue curve) better fits the original distribution (red circles) than the random-choice sampling technique (green curve). Therefore, the inverse transform sampling is better to simulate the original distribution.

3.3.2 Distribution Similarities

In this section, we describe the similarities used for transformed distributions in the previous step. In particular, we use Minkowski distance as the following equation.

$$DBS_p(a_1, a_2) = ||x_{a_1} - x_{a_2}||^{\frac{1}{p}} \tag{3.4}$$

Where a_1 and a_2 are two attributes, p is a rational number. In our experiments, we test the p on the list of $[1, 2, \infty]$ since these metrics correspond to popular similarity metric as Manhattan, Euclidean, and Chebyshev distance.

When $p = 1$ the DBS_1 metric corresponds to the Manhattan distance.

$$DBS_1(a_1, a_2) = ||x_{a_1} - x_{a_2}||_1 \tag{3.5}$$

When $p = 2$ the DBS_2 metric corresponds to the Euclidean distance.

$$DBS_2(a_1, a_2) = ||x_{a_1} - x_{a_2}||_2 \tag{3.6}$$

When $p = \infty$ the DBS_∞ metric corresponds to the Chebyshev distance.

$$DBS_\infty(a_1, a_2) = ||x_{a_1} - x_{a_2}||_\infty \tag{3.7}$$

After the similarity searching process, we have a ranking list of semantic labels ordered by their corresponding similarity scores.

3.4 Evaluation

In this section, we first describe the benchmark datasets (Section 3.4), evaluation metrics (Section 3.4.1), compared baseline approaches (Section 3.4.2), experimental setting (Section 3.4.3), and experimental results (Section 3.4.4).

Benchmark Datasets

To evaluate DBS metrics, we used two datasets i.e., City Data, Open Data. City Data is the standard data used in the previous studies [36], [2] while Open Data is newly built datasets extracted from Open Data portals. We make the datasets available in this link[2].

Table 3.1: Statistical description on numerical values per semantic label of City Data and Open Data

Dataset	m	n	# values of each labels				
			all	min	max	med	avg
City Data	30	300	192,820	40	22,510	1,130	6,427
Open Data	50	500	7,329,815	120	1,671,455	12,506	146,596

[2]EmbNum+ dataset: https://github.com/phucty/embnum

The detailed statistics of each dataset are shown in Table 3.1. The number of semantic labels in a dataset is denoted as m. The number of columns in a dataset is denoted as n. In each dataset, each semantic label has 10 columns in the same semantic labels. The columns of City Data is randomly generated using 10 partitions splitting, while the columns of Open Data are the real table columns from Open Data Portals. The number of semantic labels of the new datasets is larger than City Data, enabling rigorous comparisons between DBS and other baseline approaches.

3.4.1 Evaluation Metric

We used the mean reciprocal rank score (MRR) to measure the effectiveness of semantic labeling. The MRR score was used in the previous studies [2], [36] to measure the probability correctness of a ranking result list. Intuitively, MRR metric measures the probability of correctness of the given Q query.

$$\text{MRR} = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{\text{rank}_i} \tag{3.8}$$

where rank_i is the first correct position rank in the ranking result list.

3.4.2 Baselines

We evaluate the performance of DBS_1, DBS_2, DBS_∞ with two baseline approaches Semantic Typer [2], and DSL [36]. Semantic Typer used the KS test as the similarity metric for numerical attributes [2]. DSL used a new metric with a combination of KS Test, U Test, and the numeric Jaccard similarity.

3.4.3 Experimental Setting

In this section, we describe the detail experimental setting to evaluate the semantic labeling task. We follow the evaluation setting of Semantic Typer [2] and DSL [36]. This setting is based on cross-validation but it was modified to observe how the number of numerical values in the knowledge base will affect the performance of the labeling process. The detail of the experimental setting is described as follows.

Suppose a dataset $S = \{s_1, s_2, s_3, ..., s_d\}$ has d data sources. One data source was retained as the unknown data, and the remaining $d - 1$ data sources were used as the labeled data. We repeated this process d times, with each of the data sources used exactly once as the unknown data.

Additionally, we set the number of sources in the labeled data increasing from one source to $d - 1$ sources to analyze the effect of an increment of the number of labeled data on the performance of semantic labeling. We obtained the MRR scores and labeling times on $d \times (d - 1)$ experiments and then averaged them to produce the $d - 1$ estimations of the number of sources in the labeled data.

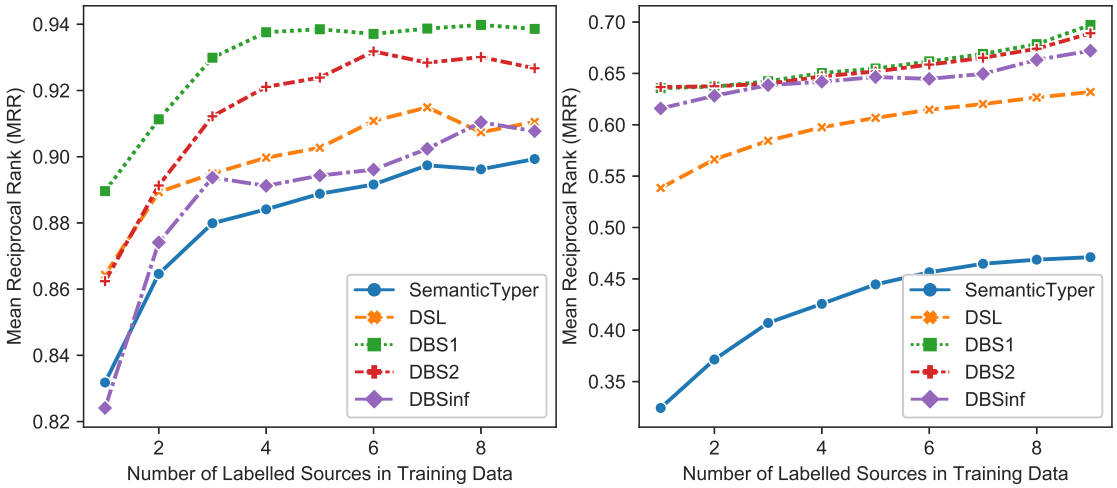
Table 3.2 depicts the semantic learning setting with 10 data sources. From 1^{st} experiment to 8^{th} experiment, s_1 is assigned as the queries of unknown sources, the remaining sources are considered as the labeled sources in the knowledge base. We conducted a similar approach for the remaining experiments. Overall, we performed 90 experiments on the 10 sources of a dataset.

Table 3.2: Semantic labeling setting with 10 data sources

Experiment	1	2	3	4	5	6	7	8	9	...	90
Queries	s_1	s_1	s_1	s_1	s_1	s_1	s_1	s_1	2	...	s_9
Knowledge Base	s_2	s_2	s_2	s_2	s_2	s_2	s_2	s_2	s_1	...	s_1
		s_3	s_3	s_3	s_3	s_3	s_3	s_3		...	s_2
			s_4	s_4	s_4	s_4	s_4	s_4		...	s_3
				s_5	s_5	s_5	s_5	s_5		...	s_4
					s_6	s_6	s_6	s_6		...	s_5
						s_7	s_7	s_7		...	s_6
							s_8	s_8		...	s_7
								s_9		...	s_8

3.4.4 Experimental Results

The results of semantic labeling for numerical values in the MRR score on City Data and Open Data is shown in Figure 3.4.



(a) City Data (b) Open Data

Figure 3.4: Semantic labeling results in the MRR score on City Data, Open Data

The MRR scores obtained by three methods steadily increase along with the number of labeled sources. It suggests that the more labeled sources in the database, the more accurate the assigned semantic labels are. DSL outperformed Semantic Typer in City Data (Figure 3.4a) and Open Data (Figure 3.4b) because it used the information from multiple testing results.

The similarity metric based on a specific hypothesis test, which was used in Semantic Typer and DSL, is not optimized for semantic meanings with various data types and distributions in general cases. The DBS metrics without using p value-based metrics outperform all baseline approaches on the two datasets (City Data and Open Data). It could be explained that in DBS metrics take consider all the differences between the empirical distribution of numerical attributes, therefore allow it more generalized than other p value metrics.

In three tested DBS metrics, the DBS_1 (Manhattan distance) achieved the highest performance in the two datasets. The level of differences between DBS metrics and other approaches is stable across the increasing number of labeled sources in training data.

The experimental results of semantic labeling systems are different on City Data and Open Data. On the City Data, all the systems achieve higher performance than the experiments on Open Data. The reason for the differences could be explained that

the Open Data contains semantic labels in multiple scaling which is more difficult than Open Data (which is extracted from normalized attributes in DBpedia). Despite the differences in difficulty level across datasets, DBS metrics also outperform other approaches.

3.5 Conclusion

In this section, we first point out the limitation of the p value based similarities, as a result, these are unstable for general cases. Then, we introduce DBS, a category of similarities derived from the norms of the inverse transform sampling of numerical attributes. The experimental results showed that DBS_1 (Manhattan distance) achieved the best performance for the task of semantic labeling for numerical values.

The current limitation of DBS is that the system can not recognize the new semantic attributes which are not available in the knowledge base. In the next section, we will describe the more advanced method called EmbNum+ to address this issue. Moreover, EmbNum+ which is a numerical attribute semantic annotation using deep metric learning achieve higher performance than DBS in the semantic annotation for numerical attributes.

4

EmbNum+: Deep Metric Learning for Numerical Attribute Annotation

In this section, we introduce EmbNum+; a deep similarity metric for numerical attribute annotation. EmbNum+ also addresses the problem of the p value-based similarities in other baseline approaches, and it improves the performance of the metric by allowing training directly on numerical attributes. Evaluations on many datasets of various domains confirmed that EmbNum+ consistently outperformed other state-of-the-art approaches in terms of accuracy. Furthermore, attribute-augmentation can be used to enhance the robustness and unlock the portability of EmbNum+, making it possible to be trained on one domain but applicable to many different domains.

We first introduce the problem and Embnum+ in Section 4.1. In Section 4.2, we define the terms, concepts, and common notations used entirely this section as well as the semantic labeling problem for numerical attributes. We present the EmbNum+ approach in Section 4.3. In Section 4.4, we describe the details of our evaluation, and experimental settings then present the results. Finally, we summarize and discuss the future direction in Section 4.5.

4.1 Introduction

Thanks to the Open Data movement, a large number of table data resources have been published on the Web or Open Data portals. For example, 233 million tables were extracted from the July 2015 version of the Common Crawl[1] [61]. Additionally, 200,000 tables from 232 Open Data portals were analyzed by Mitlohner et al. [62]. These resources could be integrated, and enabling them to be potentially useful for other applications such as table search [47, 48], table extension [49], completion[50], or knowledge base construction as used in DBpedia [51], YAGO [25], and Freebase [52].

However, these data resources are very heterogeneous. Each data resource is independently constructed by different people with different backgrounds, purposes, and contexts. Therefore, the use of vocabulary and schema structures might differ across various data resources. For example, one table attribute uses “population" as the table header label and another table attribute uses “number of people." Do these two attributes labels share the same or different meanings? This “semantic heterogeneity" may lead to the propagation of misinformation in the data integration process.

[1]Common Crawl link: http://commoncrawl.org/

To provide a unified view of the heterogeneous resources, one of the possible solutions is to assign a semantic label for each attribute in unknown resources. We categorize these semantic labeling approaches into three groups with respect to the data type: textual-based, numerical-based, and hybrid, which is a combination of the results of textual-based and numerical-based semantic labeling. The main focus of this section is numerical-based approaches.

The most common approaches for semantic labeling use textual information, such as header labels, textual values, or table description. Previous studies used text-based entity linkage to search for similar concepts and entities in knowledge bases [13, 17, 22, 51, 56, 57]. Then, semantic labels can be inferred by using rich lexical and semantic information of matched concepts and entities in the knowledge base. However, many attributes do not have overlapping entity labels with knowledge bases. Even when overlapped, many entity labels do not have similar representation with the entities in knowledge bases because these are expressed as IDs, codes, or abbreviations [5]. Additionally, the numerical values of attributes rarely have the same set of values as the relevant values in knowledge bases; therefore, it is ineffective to straightforwardly apply linguistic approaches into these numerical attributes.

In a study on profiling Open Data portals, Mitlohner et al. showed that 50 % of the table data extracted from Open Data portals contain numerical columns with missing or ambiguous headers [62]. Therefore, annotating semantic labels for these numerical columns is an important task in the data-integration procedure. Prior studies proposed general work-flows based on a retrieval setting where the label of an unknown attribute is assigned by the label of the most relevant attribute in labeled data with respect to a specific similarity metric [2, 5, 35, 36]. The most common approach is using the p value of statistical hypothesis tests as a metric to measure the similarity between lists of numerical attributes.

We argue that these p value-based metrics have a critical issue that needs to be addressed. The issue is how to choose the appropriate statistical test for hypothesis testing. In fact, the statistical hypothesis tests depend on the distribution and type of data being analyzed. For instance, these data attributes have to be drawn from a specific form of distribution (e.g., normal distribution or uniform distribution) or data types (e.g., continuous or discrete). However, determining the form of distributions and data types of unknown numerical attributes is a difficult challenge. As a result,

a proper hypothesis test cannot be easily selected when we do not know the data distribution and data type.

In recent years, deep metric learning has achieved considerable success in extracting useful representations of data and a similarity metric directly from data [63, 64, 65, 66, 67, 68]. Inspire from their success, we propose a neural numerical embedding approach, called EmbNum+, to learn a similarity metric directly from numerical attributes without the need of making an assumption regarding data type and data distribution. In particular, we use a combination of a CNN network and the triplet network, to jointly learn representations and a metric to measure the similarity between numerical attributes. Different from image data, EmbNum+ models the discrimination features from the distribution presentations of numerical attributes.

Overall, we address the limitations of prior approaches based on three dimensions: *effectiveness*, *efficiency*, and *robustness*.

Effectiveness - EmbNum+ was inspired by deep metric learning approaches with which both representations and a similarity metric are learned without making any assumption regarding data. In particular, we used a representation network consisting of a triplet network and convolutional neural network (CNN) to map numerical attributes into feature vectors in an embedding space. The “semantic similarities" of numerical attributes are calculated on these feature vectors. In other words, the CNN network learns discriminate features from numerical attributes. The triplet network approximates the goal of similarity-metric learning, which involves pulling numerical attributes with the same semantic labels into nearby positions while pushing numerical attributes with different labels apart. The distance metric is directly learned from data without the need to make any assumption regarding data type or data distribution; hence, enabling EmbNum+ to be more generalized with a variety of data types and distributions.

Efficiency - Efficiency is an important factor to take into account the task of semantic labeling because of data velocity, and data volume in the current interest in the Open Data movement. We need a lightweight approach to support real-time and large-scale semantic labeling. EmbNum+ has two advantages in terms of efficiency compared with other approaches. First, EmbNum+ is promising for efficient data storage and memory usage since all numerical attributes are stored by their representations derived from the embedding model. In fact, the size of these attributes could be very large; up to

a million of numerical values; hence, keeping entire numerical values is memory intensive. Second, EmbNum+ reduces computational overheads using the fast similarity calculation on these representations.

We also introduced an inverse transform sampling [60] to deal with the issues of varying input size of table numerical attributes. The representation network required a fixed size as the input, but the size of numerical attributes could vary from a few numerical values to thousands of them [62]. The sampling technique can capture the original distribution of numerical attributes, thereby overcoming the issue of varying the size of the input attributes. Moreover, the sampling technique also helps to speed up data processing since a small number of data values is considered instead of the entire values of attributes.

Robustness - To learn a robust EmbNum+, we introduce an attribute-augmentation technique, which is automated to generate more training data from available data. Our experiments showed that EmbNum+ is robust in all tested datasets and it can be used to learn discriminant representations and a similarity metric from a single domain data and uses across multiple domains.

Additionally, We introduce the attribute augmentation technique to generate more training data, therefore, it makes EmbNum+ more robust to over-fitting. EmbNum+ can recognize whether a query numerical attribute is a new semantic label or not. In particular, we introduce a lightweight relevance-learning approach to model the relevant information between the labeled numerical attributes. Using the relevance model can help to filter the retrieval-ranking results. If the size of the ranking result is zero, it means that the query numerical attribute is a new semantic label.

We evaluated the performance of EmbNum+ against two baseline approaches, i.e., Semantic Typer [2], and DSL [36] on four datasets: one standard dataset, e.g., City Data [2], two synthesis datasets, e.g., DBpedia Numerical Knowledge Base (NKB), WikiData NKB, and one real-world dataset, e.g., Open Data extracted from five Open Data Portals. The overall results show that EmbNum+ outperformed all other approaches in all experimental settings in terms of effectiveness, efficiency, and robustness.

4.2 Definitions and Notations

In this section, we define the mathematical notations (Section 4.2.1), then describe the problem statement of semantic labeling for numerical attributes (Section 4.2.2).

4.2.1 Notations

We refer to Table 4.1 as the mathematical notations frequently used in this section.

4.2.2 Problem Definition

The problem of semantic labeling for numerical values is defined as follows. Let $A = \{a_1, a_2, a_3, ..., a_n\}$ be a list of n numerical attributes and $Y = \{y_1, y_2, y_3, ..., y_m\}$ be a list of m semantic labels. Given (1) an unknown attribute a_q, and (2) a numerical knowledge base $D = \{(a_i, y_j)|a_i \in A, y_j \in Y\}$, where (a_i, y_j) is a data sample of a pair of a numerical attribute and its corresponding semantic label, the objective of semantic labeling is to identify the list of relevant semantic labels in D where these attributes are most likely closed to the unknown attribute a_q with respect to a numerical similarity metric.

4.3 EmbNum+ Approach

In this section, we present the overall framework of EmbNum+ in Section 4.3.1. The detail of each component is described from Section 4.3.2 to Section 4.3.5.

4.3.1 Framework

Figure 4.1 depicts the semantic labeling task with EmbNum+. The general work-flow is composed of four phases: *representation learning*, *relevance learning*, *semantic labeling offline*, and *semantic labeling online*.

The overall work-flow of semantic labeling starts with the *representation learning* as shown in Figure 4.1 (a) (Section 4.3.3) that jointly learn the discriminative representations and a similarity metric across numerical labeled attributes. Given numerical *labeled attributes*, new labeled data samples are first generated with the

Table 4.1: List of frequently mathematical notations in Chapter 4

Symbol	Description
(a_i, y_j)	a pair of numerical attribute and its corresponding semantic label, $(a_i, y_j) \in D$
a_i	a numerical attribute, $a_i \in A$
a_q	a unknown attribute
A	a list of numerical attributes
a	a numerical attribute
D	a numerical knowledge base
$emb(x)$	the embedding function for a tensor x, $emb(x) \in \mathbb{R}^k$
h	the output size of attribute transformation
k	the output size of embedding model
m	the number of semantic labels
n	the number of numerical attributes
p	the percentage probability
$cdf(v)$	the cumulative distribution functions (CDF) of a numerical value
$icdf(p)$	the inverse cumulative distribution functions of a probability p
$T_{i,j}$	a table cell
T	a table
$trans(V_a)$	the transformation for list of numerical values V_a
V_a	a list of numerical values of the attribute a
v	a numerical value
x	a tensor of numerical attributes, $x \in \mathbb{R}^h$
y_j	a semantic label, $y_j \in Y$
Y	a list of semantic labels

attribute augmentation (Section 4.3.2). Second, *attribute transformation* (The details of attribute transformation was described in Section 3.3.1) converts these augmented labeled attributes from numerical values into distribution presentations. Third, these distribution presentations will be used as the input for *representation learning*. Finally,

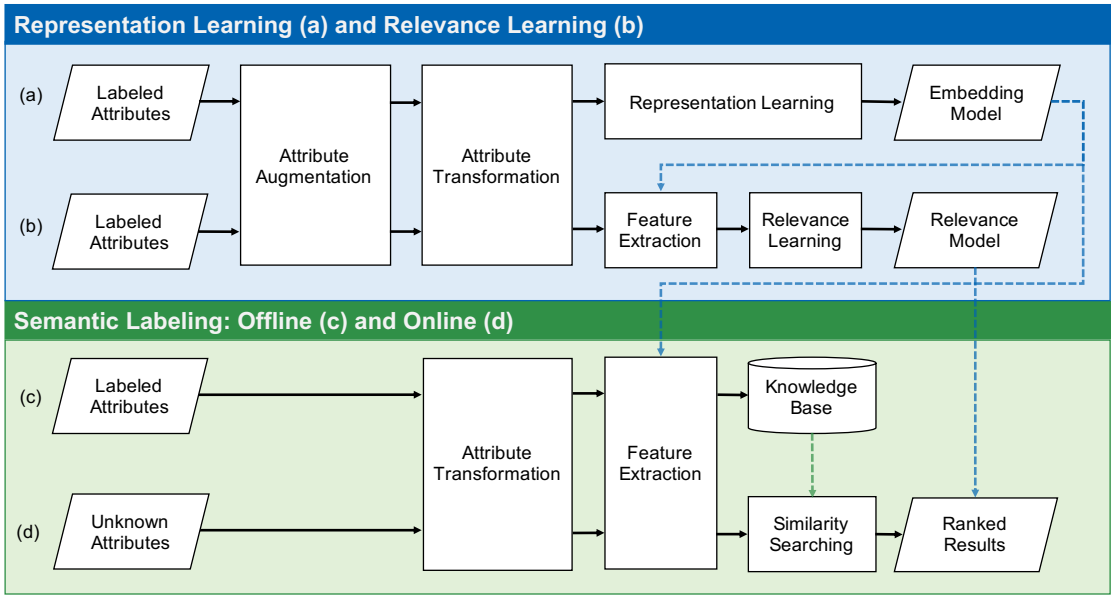


Figure 4.1: Semantic labeling framework for numerical attributes with EmbNum+ comprised of (a) representation learning, (b) relevance learning, (c) semantic labeling offline, and (d) semantic labeling online.

the output of the *representation learning* is an *embedding model* which is used as the *feature extraction* module in the later phases.

To determine whether a query attribute is a new semantic label, we introduced the *relevance learning* as shown in Figure 4.1 (b) (Section 4.3.4). Specifically, we used the logistic regression to learn a *relevance model* that predicts the relevance probabilities from pair-wise similarities of the labeled attributes. To calculate the pair-wise similarity of the labeled attributes, the *attribute augmentation*, and the *attribute transformation* are also carried out and *feature extraction* is done to derive the feature vectors using the *embedding model* learned in the representation-learning phase. The pair-wise similarities are calculated on those feature vectors. The *relevance model* will be used to filter the ranking results in the *semantic-labeling-online* phase. If the size of the ranking results is zero after filtering, the query attribute is a new semantic label.

As mentioned above, there are two *semantic-labeling* phases as shown in Figure 4.1 (c) (Section 4.3.5): *offline* and *online* (hereafter, offline phase and online phase, respectively). The *offline phase* involves data preparation while the *online phase* is actually semantic labeling for an unknown attribute. In the *offline phase*, *labeled*

attributes are standardized with attribute transformation and derived feature vectors with feature extraction. These feature vectors are stored in the *knowledge base* for future similarity comparison.

In the *online phase* of semantic labeling as shown in Figure 4.1 (d), the first two steps are similar to those of the *offline phase* where an unknown attribute is standardized with *attribute transformation* and feature vectors are derived with *feature extraction*. Then, the *similarity searching* module is used to calculate the similarities between the feature vector of the unknown attribute with all the feature vectors in the *knowledge base*. After this process, we have a ranking list of semantic labels ordered by their corresponding similarity scores. Then, the *relevance model* is used on this ranking list to select only the relevant semantic labels based on these scores. Finally, the output is a ranking list of the most relevant attributes.

In the following sections, we describe the details of each component in the overall work-flow starting with the attribute augmentation, then, attribute transformation, the two learning phases: representation learning and relevance learning, and the two semantic-labeling phases: online and offline. Readers may refer to Section 4.2.1 for the explanation of mathematical notations and equations.

4.3.2 Attribute Augmentation

In this section, we introduce a technique for augmenting attributes from originally labeled attributes to create more labeled attributes for learning. In principle, we need many labeled attributes to learn a robust embedding model and relevance model; however, the lack of labeled data is a common problem with this task. The size of datasets for this task is not so large, for example, the City Data [2] contains 300 numerical columns with 30 semantic labels, and the Open Data [69] contains 500 numerical columns with 50 semantic labels. The reason for this issue is that it is very time consuming and costly to manually assign the semantic labels for table data. Therefore, we need to augment existing data to increase data size.

To address this issue, we introduced an attribute-augmentation technique based on the intuition that the semantic label of an attribute does not change whether we have several or thousands of numerical values. Therefore, we can generate samples by changing the size of attributes and randomly choosing the numerical values in

the original attributes. While using the attribute-augmentation technique for an attribute, the distribution of augmented attributes slightly different than the original one. Therefore, it increases the variety of attribute distribution and addresses the issue of lacking training data.

Figure 4.2 depicts an example of attribute augmentation for the attribute *height*. The input is the original attribute (Left), and the output is the augmented attributes (Right). We use all data (original and augmented attributes) during the training process.

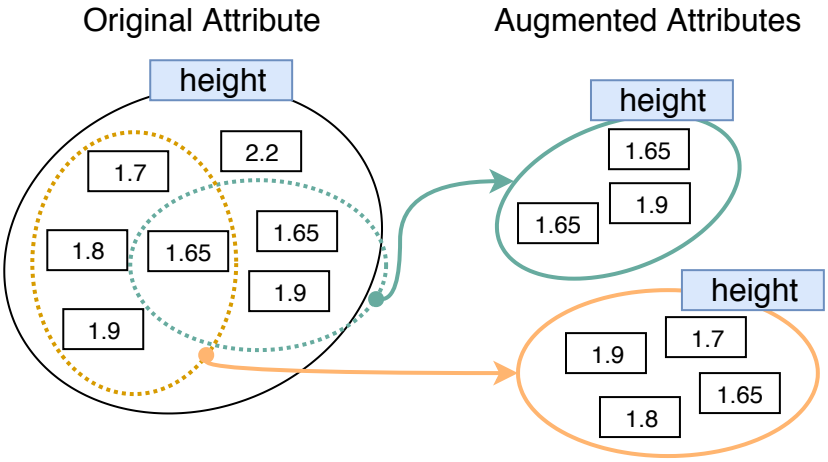


Figure 4.2: Illustration of augmentation on a numerical attribute.

The attribute-augmentation technique is described as follows. Given (1) an attribute a having n numerical values $V_a = [v_1, v_2, v_3, ..., v_n]$, and (2) a number of new data samples *aug_size*, this attribute-augmentation technique first randomly determines a new size $n' \in [min\_size, n]$ for the new sample, then randomly selects n' values from V_a to V'_a, where $V'_a \subseteq V_a$. The process is repeated *aug_size* times to create new *aug_size* samples for a.

Figure 4.3 illustrates an example of attribute augmentation for the numerical attribute of *areaLandSqMi* in City Data [2]. The blue line represents 8327 original values, and dotted lines represent the augmented samples from original values. The figure depicts the probability distribution of the original values of *areaLandSqMi* (blue line) and five augmented samples (dotted lines). The five augmented samples vary in terms of attribute size and shape, increasing the variety of training data, and reducing the overfitting for representation learning and relevance learning.

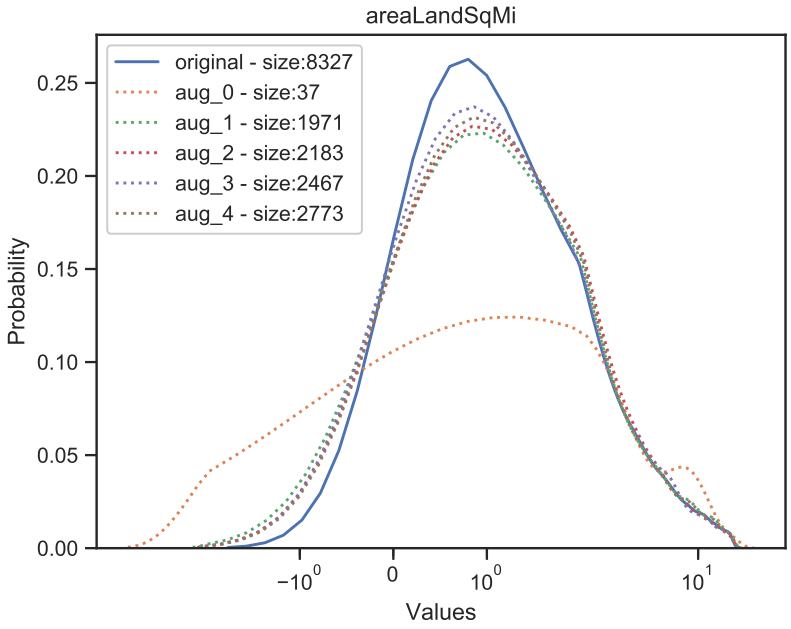


Figure 4.3: Attribute augmentation of attribute of *areaLandSqMi* in City Data [2].

4.3.3 Representation Learning

In this section, we describe the representation-learning phase (Figure 4.4) to learn the embedding models.

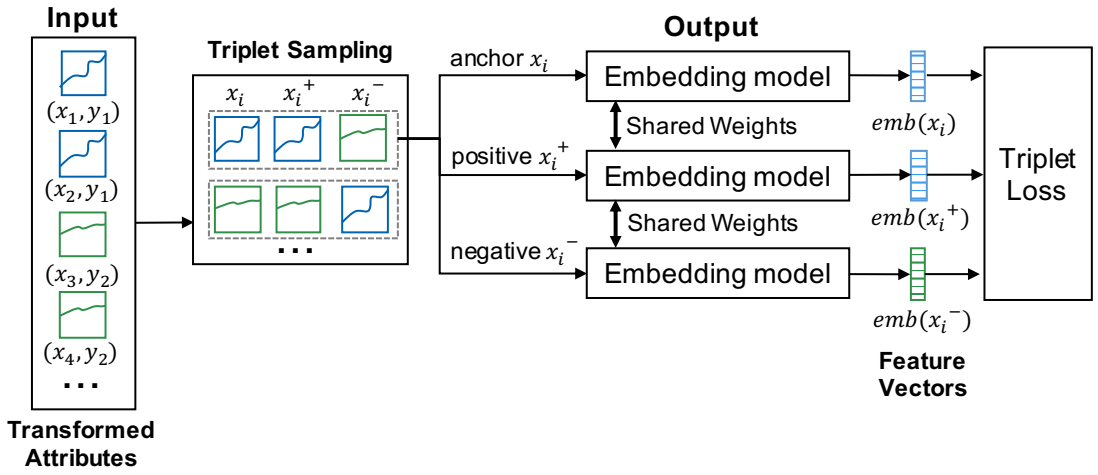


Figure 4.4: Representation-learning architecture

Given an list of n transformed attribute $X = \{x_1, x_2, x_3, ..., x_n\}$ and their m semantic labels $Y = \{y_1, y_2, y_3, ..., y_m\}$, we first performed triplet sampling to create a list of triplets. A triplet (x, x^+, x^-) with $(x, x^+, x^- \in X)$ is a combination of an anchor x where the semantic label is y, a positive attribute x^+ where the semantic label is y, and a negative attribute x^- where the semantic label is not y. In the Figure 4.4, the blue square, and green square indicate the semantic label y_1 and the semantic label y_2, respectively, where $y_1 \neq y_2$. We used the hard negative sampling method [67] (negative samples close to anchor) to select triplets for training. This method involves choosing the closest sample to an anchor among the dissimilar attributes in a mini-batch of learning and helps the training process to converge faster.

We used the triplet network and CNN to learn a mapping function $emb(\cdot)$ for numerical attributes of triplets into an embedding space so that the distance between a positive pair must be less than that between a negative pair $d_{emb}(x, x^+) < d_{emb}(x, x^-)$ [67]. Recall a numerical attribute x, $emb(x)$ is the output of a representation learning network to convert x into a k dimensions embedding space $emb(x) \in \mathbb{R}^k$. We used the two-dimensional Euclidean plane as the embedding space because it is the most common space used in the literature. It is noticed that we also can use any of other spaces as the embedding space. The distance between two numerical attributes x_i and x_j is calculated by using the Euclidean distance between $emb(x_i)$ and $emb(x_j)$:

$$d_{emb}(x_i, x_j) = d(emb(x_i), emb(x_j)) = ||emb(x_i) - emb(x_j)||_2^2 \tag{4.1}$$

The triplet loss function for the triplet network is defined as follows.

$$L = max(0, \alpha + d_{emb}(x, x^+) - d_{emb}(x, x^-)) \tag{4.2}$$

where α is a hyper-parameter that regularizes between positive pair distance and negative pair distance.

Embedding model with CNN architecture We use a CNN architecture to learn the embedding model since it can capture latent features directly from data. Instead of designing a CNN architecture from scratch, we adopted the popular CNN architectures such as VGG [70] and ResNet [71] architectures. Since the input data were one-dimensional, we modified the convolutional structure to be one-dimensional on the

convolutional, batch normalization, and pooling layers. The output of the CNN is a vector with k dimensions (Section 4.4.3).

Table 4.2 depicts a comparison of different CNN architectures on model size, and model depth of the adapted VGG and ResNet (one-dimensional data). Intuitively, the more layers of the network have, the better information capacity model can carry [71]. Finally, we came up with the ResNet 18 since it requires fewer parameters but deeper comparing to other networks. It is noticed that we adopted the Resnet 18 as a practical reason, we can use other CNN architectures instead. The optimal designing CNN architectures are left as future work.

Table 4.2: Comparison of CNN architectures in terms of size, and depth

CNN Networks	**# parameters**	**# layers**
ResNet 18	3,854,164	18
ResNet 34	7,228,500	34
ResNet 50	15,995,220	50
ResNet 101	28,302,676	101
ResNet 152	38,441,300	152
VGG 11	34,628,436	11
VGG 13	34,690,452	13
VGG 16	36,463,764	16
VGG 19	38,237,076	19

4.3.4 Relevance Learning

In this section, we describe the relevance-learning architecture in details. Figure 4.5 illustrates the procedure of relevance learning.

Given the list of n transformed attributes $X = \{x_1, x_2, x_3, ..., x_n\}$ and their m semantic labels $Y = \{y_1, y_2, y_3, ..., y_m\}$, we first used the learned embedding model to derive the feature vectors for those attributes (apply mapping function $emb(\cdot)$ on each transformed attribute $x \in X$). We then carried out the pair-wise similarities (all combination) across attributes with the Euclidean distance on their embedding vectors

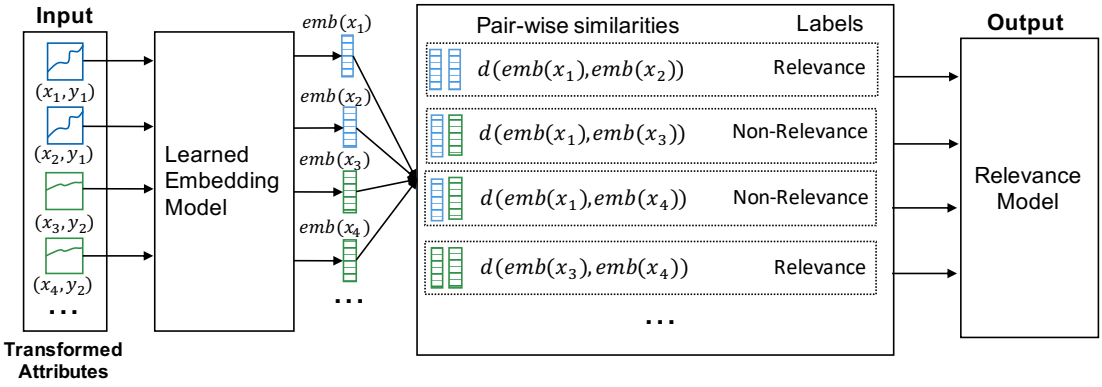


Figure 4.5: Relevance learning architecture

(Equation 4.1). Each pair-wise similarity is used as an input variable, while its output label (relevance or non-relevance) is determined by the semantic labels. If two attributes have the same semantic label, the data label of the pair-wise similarity is relevance, else the label is non-relevance. All pairs of input variables and output labels are used as training data for the relevance model.

We used the logistic regression (binary classification) to learn the *relevance model*. It is noticed that we can use other binary classification algorithms. Regarding the logistic regression, a sigmoid function is used to map the pair-wise similarities to probabilities as Equation 4.3.

$$\sigma(d) = \frac{1}{1 + e^{-(w_1 * d + w_0)}} \tag{4.3}$$

where w_1, w_0 are the learning parameters for logistic regression model, d is a pair-wise similarity. We used θ as the decision boundary parameter. If the probability $\sigma \geq \theta$, the prediction is relevance, else the prediction is non-relevance.

4.3.5 Semantic Labeling

The *semantic labeling* have two phases: *offline* and *online*. The *offline phase* involves data preparation while the *online phase* involves actual semantic labeling for an unknown attribute.

Offline phase

In the *offline phase*, *labeled attributes* are standardized with attribute transformation (Section 3.3.1) and feature vectors are derived with feature extraction (Section 4.3.3). These feature vectors are stored in the *knowledge base* for future similarity comparisons.

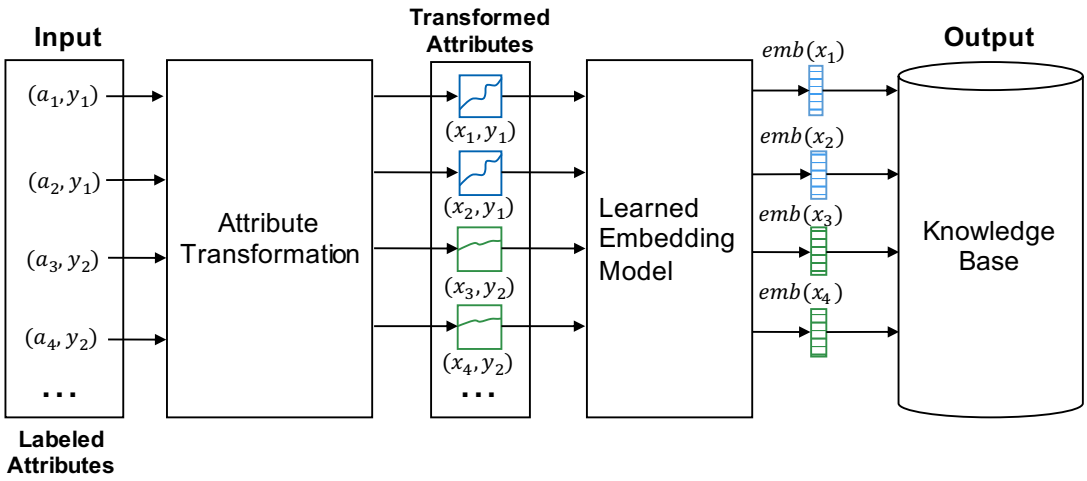


Figure 4.6: Offline-semantic-labeling architecture

Figure 4.6 depicts an example of semantic labeling for an unknown attribute. Given labeled attributes and their semantic labels $\{(a_1, y_1), (a_2, y_2), (a_3, y_3), ..., (a_n, y_m)\}$, the data is first transformed into $\{(x_1, y_1), (x_2, y_2), (x_3, y_3), ..., (x_n, y_m)\}$ using the attribute-transformation module. After that, the mapping function $emb(.)$ is used to map all the labeled data into the embedding space $\{(emb(x_1), y_1), (emb(x_2), y_2), (emb(x_3), y_3), ..., (emb(x_n), y_m)\}$. These data are stored in a knowledge base D for the next comparison in the semantic-labeling online phase.

Online phase

In this section, we describe the Online phase of semantic labeling for unknown attributes. Figure 4.7 depicts the process of the online phase. Given an unknown attribute a_q, the two first steps are similar to those of the *offline phase* where an unknown attribute is standardized to x_q with *attribute transformation* (Section 3.3.1) and $emb(x_q)$ features are derived with *feature extraction* (Section 4.3.3). Then, the *similarity-searching* module is used to calculate the similarities (Equation 4.1) between

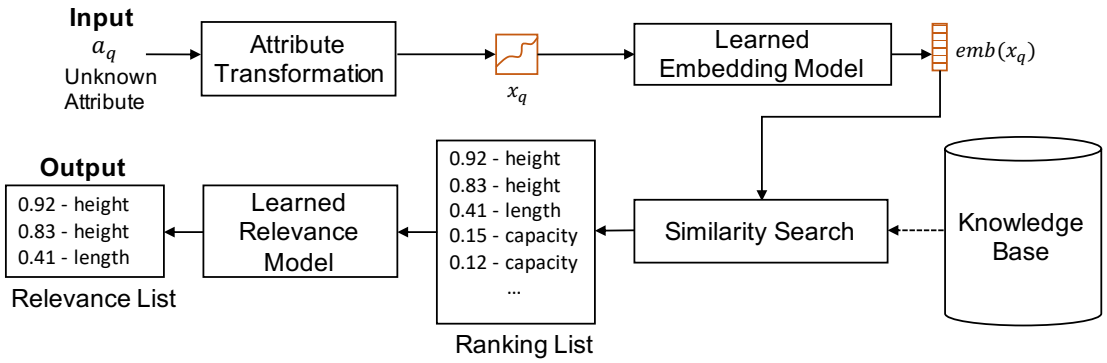


Figure 4.7: Online-semantic-labeling architecture

the feature vector of the unknown attribute with all the feature vectors in the *knowledge base D*. After this process, we have a ranking list of semantic labels ordered by their corresponding similarity scores. Then, the *relevance model* is used on those similarity scores to predict the semantic labels is relevance or non-relevance using Equation 4.3 (Section 4.3.4). Then, we remove all the non relevance semantic labels from the ranking list. Finally, the output is a ranking list of the most relevant attributes.

4.4 Evaluation

In this section, we first describe the benchmark datasets in Section 4.4.1 and evaluation metrics in Section 4.4.2. Next, we present the implementation details and experiment settings in Section 4.4.3. Finally, we explain the results and detail analysis on semantic labeling for numerical values and of in Section 4.4.4.

4.4.1 Benchmark Datasets

To evaluate EmbNum+, we used four datasets i.e., City Data, Open Data, DBpedia NKB, and Wikidata NKB. City Data is the standard data used in the previous studies [36], [2] while Open Data, DBpedia NKB, and Wikidata NKB are newly built datasets extracted from Open Data portals, DBpedia and Wikidata, respectively. The datasets are available at https://github.com/phucty/embnum.

The detailed statistics of each dataset are shown in Table 4.3. m denotes the number

Table 4.3: Statistical description of numerical values per semantic label on City Data, Open Data, Wikidata NKB, and DBpedia NKB

Dataset	m	n	# values of each labels				
			all	min	max	med	avg
City Data	30	300	192,820	40	22,510	1,130	6,427.33
Open Data	50	500	7,329,815	120	1,671,455	12,506	146,596.3
Wikidata NKB	169	1690	5,762,389	55	1,535,812	1,055	34,096.98
DBpedia NKB	203	2030	4,467,716	56	955,957	1,632	22,008.45

of semantic labels in a dataset. n denotes the number of columns in a dataset. In each dataset, each semantic label has 10 columns in the same semantic labels. The columns of City Data, DBpedia NKB, and Wikidata NKB are randomly generated using 10 partitions splitting, while the columns of Open Data are the real table columns from Open Data Portals. The number of semantic labels of the new datasets is larger than City Data, enabling rigorous comparisons between EmbNum+ and other baseline approaches.

Table 4.4 reports the overall quantile ranges of the four datasets. DBpedia NKB is the most complex dataset in terms of the largest semantic labels (206 semantic labels) as well as the range of numerical values (the range of $[-10e10, 10e16]$). Moreover, the overlapping rate of numerical attributes in DBpedia NKB is also higher than in other datasets.

DBpedia NKB and City Data have the same source of data as DBpedia. Therefore, there is a high overlapping of attributes between these data. The two other datasets of Wikidata NKB and Open Data are different from DBpedia NKB and City Data. Wikidata NKB is constructed from Wikidata; it is an independent project manually annotates by the community. The source of Wikidata NKB is different from Wikipedia where DBpedia extracted from, therefore Wikidata NKB and DBpedia NKB are different. Open Data extracted from five Open Data Portals which are different about the domain of data with other datasets.

Table 4.4: Quantile ranges of City Data, Open Data, DBpedia NKB, and Wikidata NKB

Dataset	from	to
City Data	−10e2	10e8
Open Data	−10e6	10e14
Wikidata NKB	−10e5	10e12
DBpedia NKB	−10e10	10e16

City Data

City Data [2] has 30 numerical properties extracted from the city class in DBpedia. The dataset consists of 10 sources; each source has 30 numerical attributes associated with 30 data properties. The Figure 4.8 is shown the distributions of quantile ranges of numerical attributes of City Data.

Open Data

Open Data has 50 numerical properties extracted from the tables in five Open Data Portals. We built the dataset to test semantic labeling for numerical values in the open environment.

To build the dataset, we extracted table data from five Open Data portals, i.e., Ireland (data.gov.ie), the UK (data.gov.uk), the EU (data.europa.eu), Canada (open.canada.ca), and Australia (data.gov.au). First, we crawled CSV files from the five Open Data portals and selected files whose size is less than 50 MB. Then, we analyzed tables in CSV files and selected only numerical attributes. After that, we created attribute categories based on the clustering of the numerical attributes with respect to the textual similarity of column headers. A category contains many numerical columns with the same semantic labels. We got 7,496 categories in total.

We manually evaluated these categories with two criteria: (1) The first criterion was to pick up categories with a certain frequency. By examining the collection of data, we found that high-frequency and low-frequency categories are often unclear on their semantics. We decided to pick up the categories with ten attributes by following the setting of City Data. (2) The second criterion was removing the categories where column headers had too general meanings such as “ID," “name," or “value."

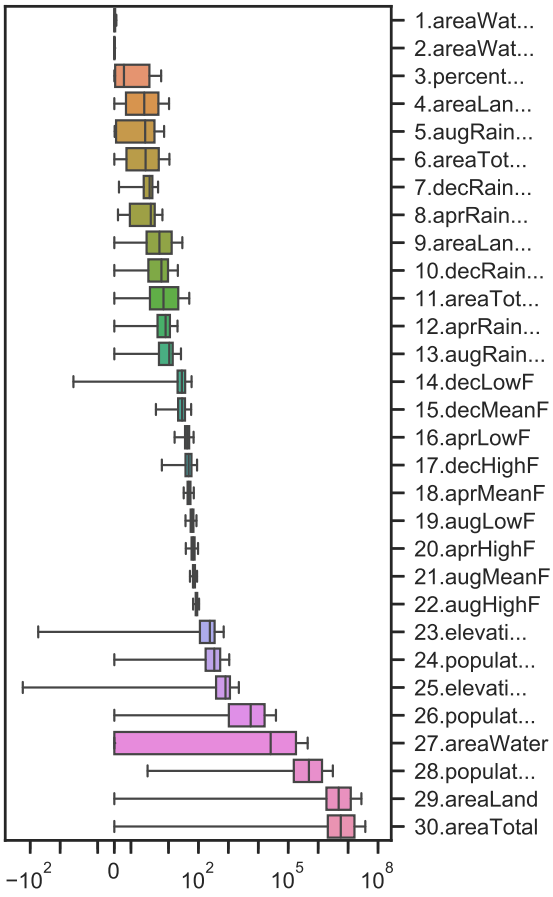


Figure 4.8: Quantile range distribution of numerical attributes in City Data

Finally, we chose 50 categories as semantic labels; each semantic label had ten numerical attributes. Following the guideline of City Data, we also made 10 data sources by combining each numerical attribute from each category.

Figure 4.9 is shown the distributions of quantile ranges of numerical attributes of Open Data.

Wikidata NKB

The Wikidata NKB was built from the most frequently used numerical properties of Wikidata. At the time of processing, there were 477 numerical properties [2] but we only selected 169 numerical properties which are used more than 50 times in Wikidata.

[2]Wikidata query service: https://query.wikidata.org/

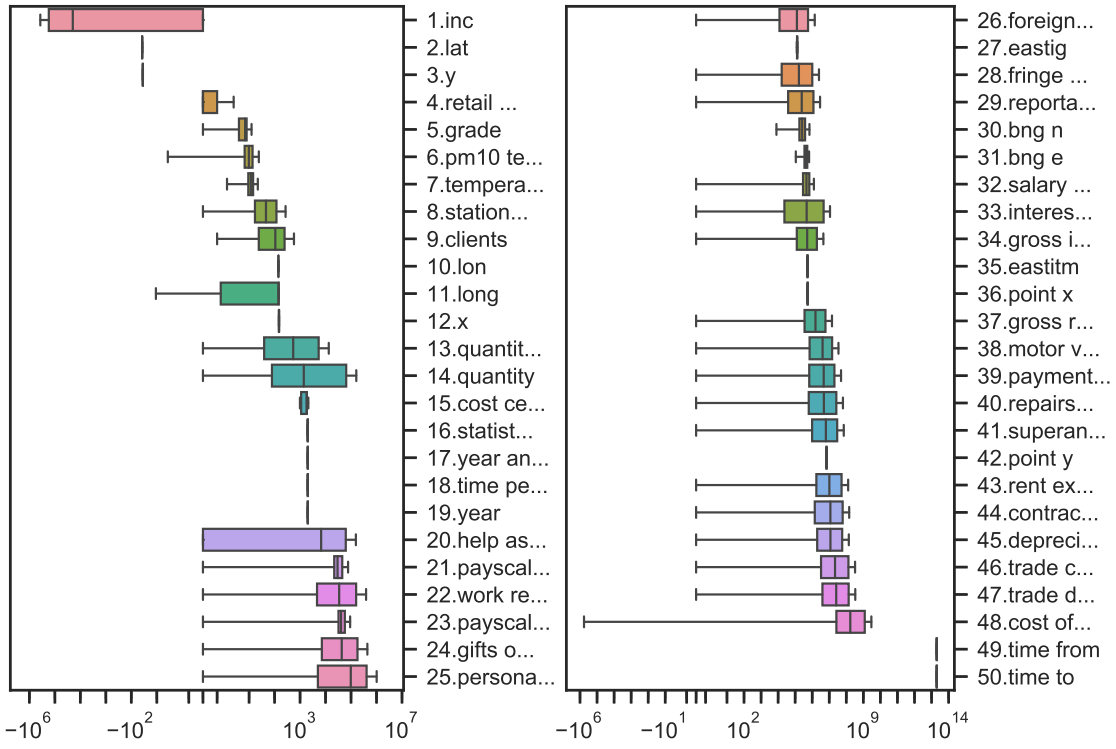


Figure 4.9: Quantile range distribution of numerical attributes in Open Data.

Figure 4.10 is shown the distributions of quantile ranges of numerical attributes of Wikidata NKB.

DBpedia NKB

To build the DBpedia NKB, we collected numerical values of the 634 of DBpedia properties directly from their SPARQL query service[3]. Finally, we obtained 206 of the most frequently used numerical properties of DBpedia where each attribute has at least 50 values.

Figure 4.11 is shown the distributions of quantile ranges of numerical attributes of DBpedia NKB.

[3]DBpedia query service: http://live.dbpedia.org/sparql

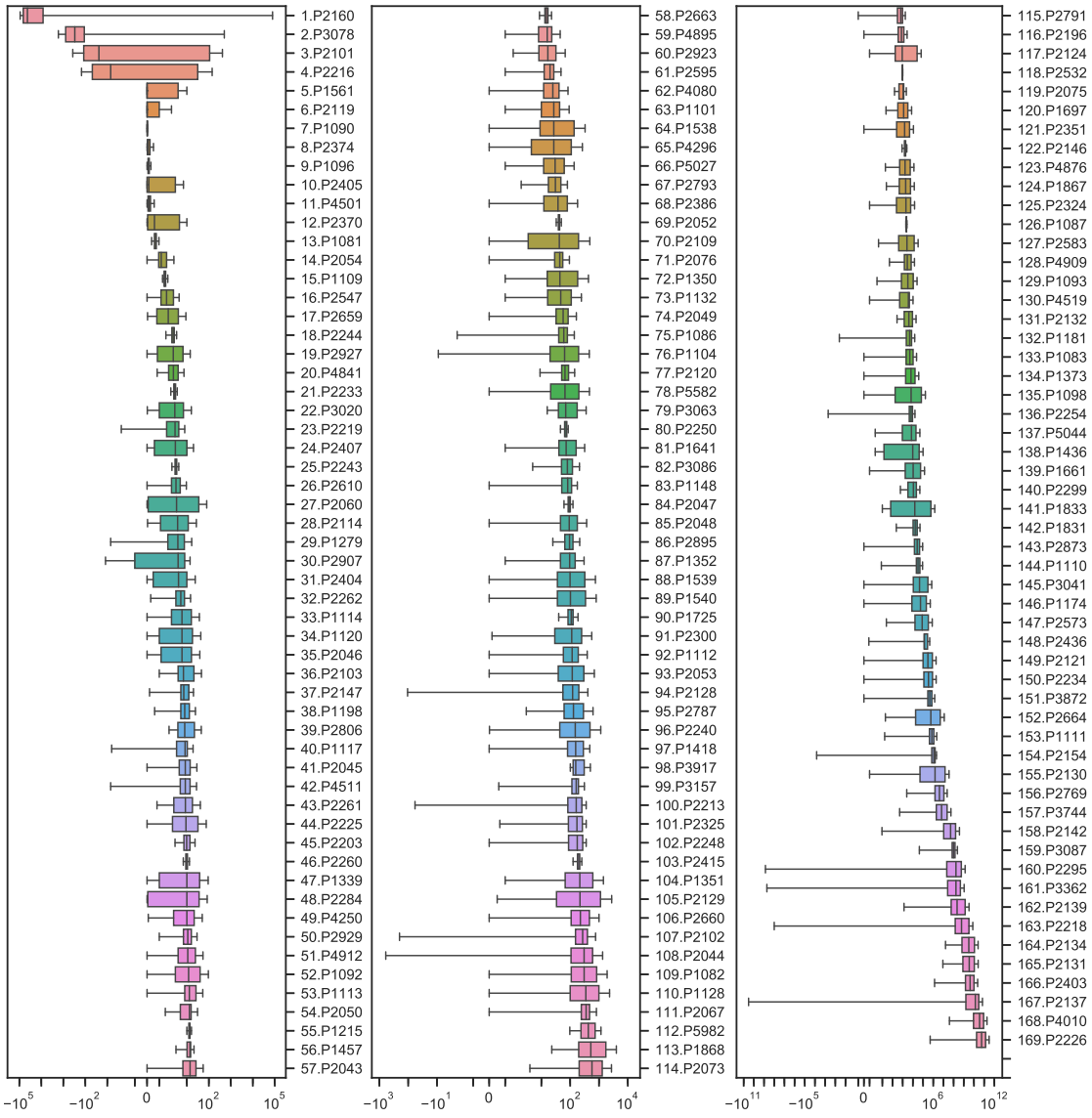


Figure 4.10: Quantile range distribution of numerical attributes in Wikidata NKB

4.4.2 Evaluation Metric

We used the mean reciprocal rank score (MRR) to measure the effectiveness of semantic labeling. The MRR score was used in the previous studies [2], [36] to measure the probability correctness of a ranking result list. To measure the efficiency of EmbNum+ over the baseline methods, we evaluated the run-time in seconds of the semantic

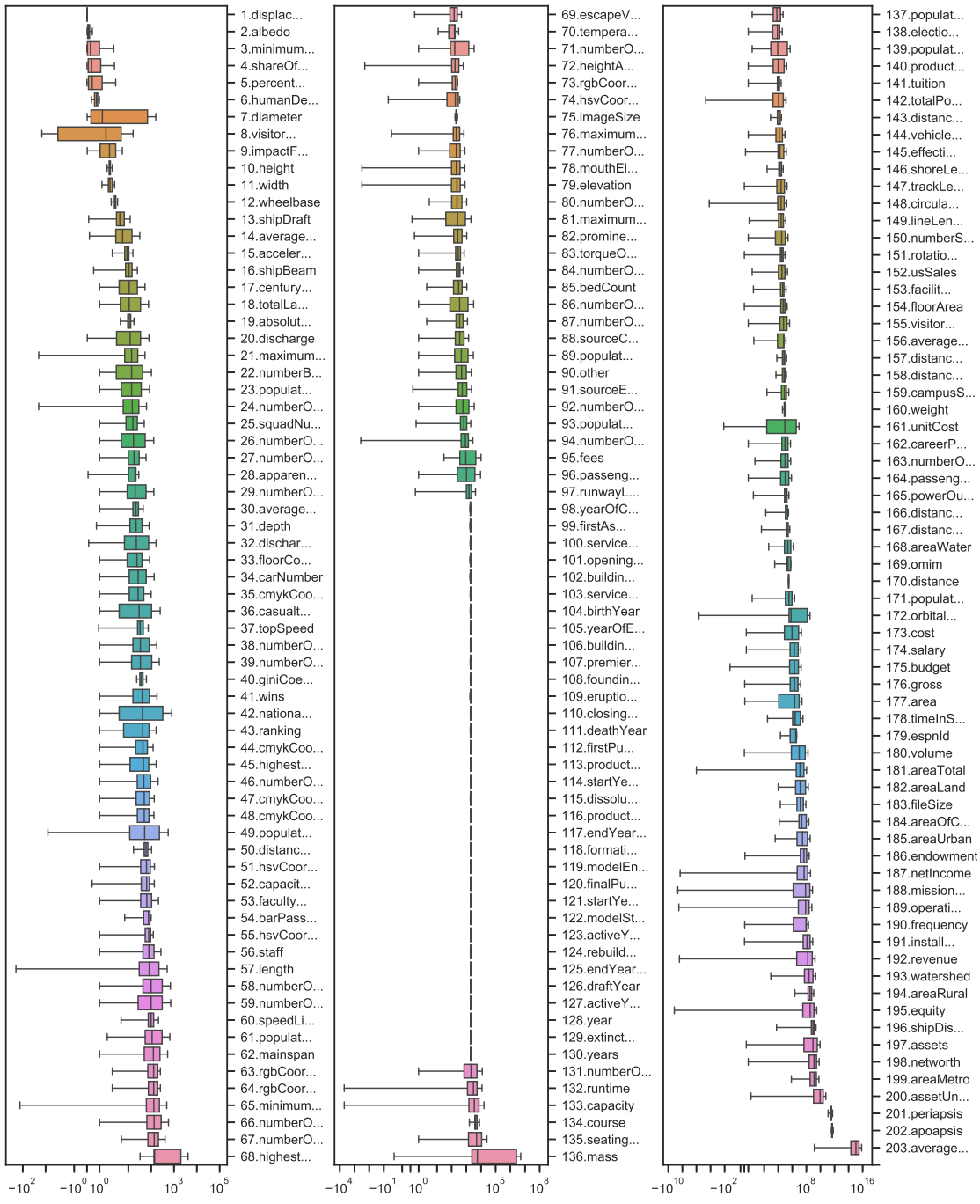


Figure 4.11: Quantile range distribution of numerical attributes in DBpedia NKB

labeling process.

4.4.3 Implementation and Settings

We have different interests in each dataset to evaluate the performance of EmbNum+. The DBpedia NKB is the most complex and complete dataset with the largest semantic labels as well as a wide range of values. It provides a discriminative power to train the representation model as well as the relevance model EmbNum+. Therefore, we use the DBpedia NKB dataset for these two learning modules. The details of the learning settings are described in Section 4.4.3. We use City Data as the standard data to fairly compared with other existing approaches. The Wikidata NKB is challenging in terms of large scale and transfer capacity setting where the embedding model is learned from DBpedia NKB. Finally, Open data is used to evaluate the real-world setting where numerical attributes are extracted from the five Open Data Portals.

Representation and Relevance Learning

To train EmbNum+, we used the numerical attributes of DBpedia NKB as the training data. We randomly divided DBpedia NKB into two equal parts: 50% for the two learning tasks and 50% for evaluating the task of semantic labeling. The first part was used for the representation learning of EmbNum+. It is noticed that we made using the attribute augmentation technique to generate training samples. Therefore, the actual training data is not the same as the original data. We also use this part to train the relevance model by using the pair-wise distance between these original training samples. This data was also used to learn the similarity metric for DSL. We followed the guideline that using logistic regression to train the similarity metrics where training samples are the pairs of numerical attributes [36].

We used PyTorch [4] to implement representation learning. The network uses the rectified linear unit (ReLU) as a non-linear activation function. To normalize the distribution of each input feature in each layer, we also used batch normalization [72] after each convolution and before each ReLU activation function. We trained the network using stochastic gradient descent (SGD) with back-propagation, a momentum of 0.9, and a weight decay of 1e−5. We started with a learning rate of 0.01 and reduced

[4]PyTorch link: http://pytorch.org

it with a step size of 10 to finalize the model. We set the dimension of the attribute input vector h and the attribute output vector k as 100.

We trained the EmbNum+ with 20 iterations. In each iteration, we used the attribute-augmentation technique to generate a *aug_size* of 100 samples for each semantic label. The numerical values of the augmented samples are randomly selected from the list of numerical values of the attributes. The size of each augmented sample ranges from *min_size* of 4 to the size of its original attribute. Then, the representation learning was trained with 100 epochs. After each epoch, we evaluated the task of semantic labeling on the MRR score using the original training data. We saved the learned model having the highest MRR score. All experiments ran on Deep Learning Box with an Intel i7-7900X-CPU, 64 GB of RAM, and three NVIDIA GeForce GTX 1080 Ti GPU.

The training time of EmbNum+ is 29,732 seconds while the training time of DSL is 2,965 seconds. It is clear that EmbNum+ uses the deep learning approach and needs more time to train the similarity metric than DSL, which uses logistic regression. However, the similarity metric is only needed to train once, and it could be applied to other domains without retraining. The details experimental result on EmbNum+ robustness is reported in Section 4.4.4.

Semantic Labeling

In this section, we describe the detail experimental setting to evaluate the semantic labeling task. We follow the evaluation setting of Semantic Typer [2] and DSL [36]. This setting is based on cross-validation but it was modified to observe how the number of numerical values in the knowledge base will affect the performance of the labeling process. The detail of the experimental setting is described as follows.

Suppose a dataset $S = \{s_1, s_2, s_3, ..., s_d\}$ has d data sources. One data source was retained as the unknown data, and the remaining $d - 1$ data sources were used as the labeled data. We repeated this process d times, with each of the data sources used exactly once as the unknown data.

Additionally, we set the number of sources in the labeled data increasing from one source to $d - 1$ sources to analyze the effect of an increment of the number of labeled data on the performance of semantic labeling. We obtained the MRR scores and

labeling times on $d \times (d-1)$ experiments and then averaged them to produce the $d-1$ estimations of the number of sources in the labeled data.

Unseen Semantic Labeling

In this section, we describe the setting of unseen semantic labeling. We split data to d partitions and used the setting of d-fold cross-validation for evaluation. To analyze the changing of EmbNum+ performance regarding the number of unseen semantic labels, we linearly increased the percentage of unseen semantic labels from 0 % to 90% of all labels in knowledge bases. For details, Table 4.5 depicts the number of unseen semantic labels of City Data, Open Data, DBpedia NKB, and Wikidata NKB.

Table 4.5: Unseen semantic labeling setting on DBpedia NKB, City Data, Wikidata NKB, and Open Data

Dataset	**0%**	**10%**	**20%**	**30%**	**40%**	**50%**	**60%**	**70%**	**80%**	**90%**
DBpedia NKB	0	20	40	60	81	101	121	142	162	182
City Data	0	3	6	9	12	15	18	21	24	27
Wikidata NKB	0	16	33	50	67	84	101	118	135	152
Open Data	0	5	10	15	20	25	30	35	40	45

The performance is evaluated with the MRR score on the four datasets. When a query is an unseen attribute, the reciprocal rank (RR) is 1 if the ranking result is empty, and 0 otherwise.

Ablation Study

We also conducted ablation studies to evaluate the impact of the representation leaning and the attribute-augmentation on the task of semantic labeling. For the setting of EmbNum+ without using the representation learning, we created three methods that ignore the representation leaning: Num_l1, Num_l2, and $Num\_l\infty$. The similarities between numerical attributes are directly calculated from the $tran(.)$ without using the embedding model. The Num_l1 used the Manhattans distance, Num_l2 used the Euclidean distance, and $Num\_l\infty$ used Chebyshev distance. For the setting of

EmbNum+ without using the attribute-augmentation, we call this method as EmbNum+ NonAu.

We conducted this ablation study based on d-fold cross-validation. Given a dataset with d data sources. One data source was retained as the query set, and the remaining $d - 1$ data sources were used as a knowledge base. We conducted semantic labeling for the query set on the knowledge base. We repeated this process d times, with each of the data sources used exactly once as the unknown data.

4.4.4 Experimental Results

In this section, we report the experimental results of semantic labeling in terms of effectiveness, robustness (Section 4.4.4), and efficiency (Section 4.4.4). Section 4.4.3 reports the experimental result of the setting of unseen semantic labeling. Finally, we report the result of the ablation study in Section 4.4.3.

Semantic Labeling: Effectiveness

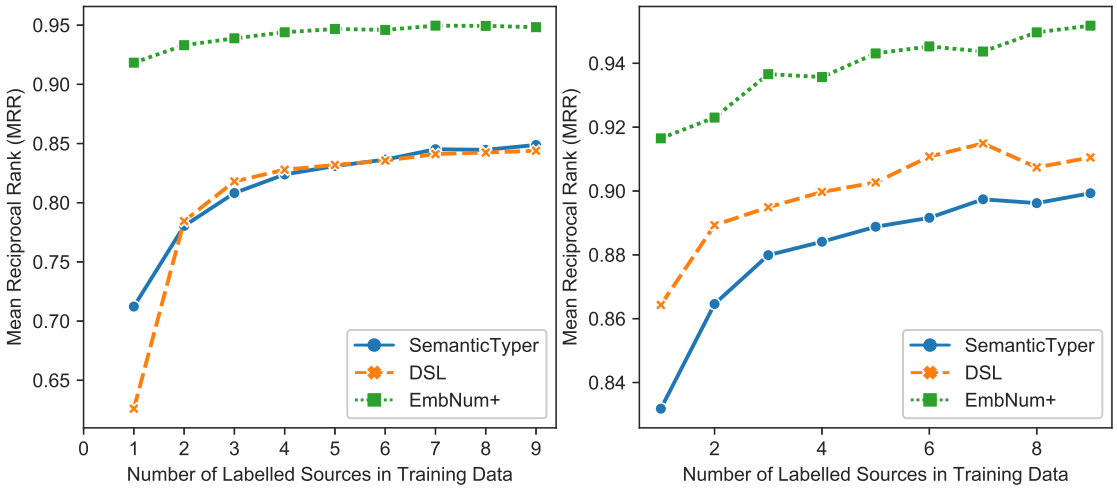
We tested Semantic Typer [2], DSL [36], and EmbNum+ on the semantic labeling task using the MRR score to evaluate the effectiveness. The results are shown in Table 4.6 and Figure 4.12.

The MRR scores obtained by three methods steadily increase along with the number of labeled sources. It suggests that the more labeled sources in the database, the more accurate the assigned semantic labels are. DSL outperformed Semantic Typer in City Data and Open Data but was comparable with Semantic Typer in DBpedia NKB and Wikidata NKB. In the DBpedia NKB and Wikidata NKB, there are more semantic labels as well as a high level of range overlapping between numerical attributes, therefore, these features (KS Test, numerical Jaccard, and MW test) proposed by DSL become less effective.

EmbNum+ learned directly from the empirical distribution of numerical values without making any assumption on data type and data distribution, hence, outperformed Semantic Typer and DSL on the four datasets. The similarity metric based on a specific hypothesis test, which was used in Semantic Typer and DSL, is not optimized for semantic meanings with various data types and distributions in DBpedia NKB and Wikidata NKB.

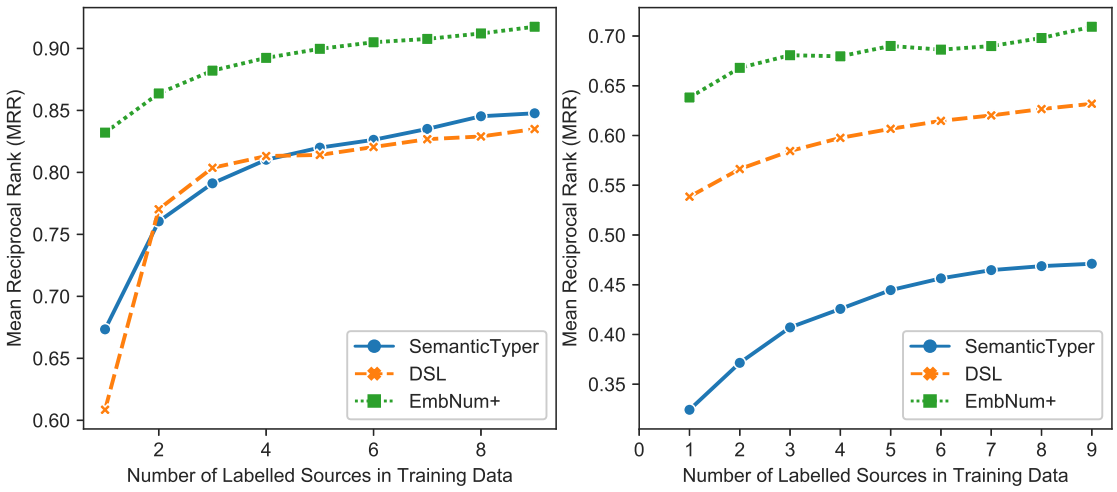
Table 4.6: Semantic labeling results in the MRR score on City Data, Open Data, DBpedia NKB, and Wikidata NKB

Dataset	Method	Number of Sources in Labels Sources in Training Data								
		1	2	3	4	5	6	7	8	9
DBpedia NKB	Semantic Typer [2]	0.7123	0.7804	0.8082	0.8239	0.831	0.8364	0.8452	0.8447	0.8488
	DSL [36]	0.626	0.7845	0.818	0.8279	0.8319	0.8356	0.8411	0.8424	0.8439
	EmbNum+ [7]	**0.9183**	**0.9331**	**0.9389**	**0.944**	**0.9468**	**0.946**	**0.9496**	**0.9494**	**0.9483**
City Data	Semantic Typer [2]	0.8318	0.8646	0.8799	0.8841	0.8888	0.8916	0.8974	0.8962	0.8993
	DSL [36]	0.8643	0.8893	0.8949	0.8997	0.9027	0.9108	0.9149	0.9074	0.9105
	EmbNum+ [7]	**0.9165**	**0.923**	**0.9366**	**0.9357**	**0.9431**	**0.9453**	**0.9437**	**0.9497**	**0.9518**
Wikidata NKB	Semantic Typer [2]	0.6734	0.7605	0.7912	0.8102	0.82	0.8263	0.8351	0.8453	0.8477
	DSL [36]	0.6085	0.7703	0.8037	0.8132	0.8141	0.8205	0.8268	0.829	0.835
	EmbNum+ [7]	**0.8321**	**0.8637**	**0.8821**	**0.8924**	**0.8997**	**0.905**	**0.9077**	**0.9121**	**0.9175**
Open Data	Semantic Typer [2]	0.3243	0.3715	0.4071	0.4257	0.4446	0.4564	0.4647	0.4687	0.4711
	DSL [36]	0.5385	0.5663	0.5844	0.5976	0.6067	0.6147	0.6202	0.6266	0.6319
	EmbNum+ [7]	**0.6382**	**0.668**	**0.6807**	**0.6796**	**0.69**	**0.6863**	**0.6899**	**0.698**	**0.7093**



(a) DBpedia NKB

(b) City Data



(c) Wikidata NKB

(d) Open Data

Figure 4.12: Semantic labeling results in the MRR score on City Data, Open Data, DBpedia NKB, and Wikidata NKB

EmbNum+ performance is slightly better than other approaches when the number of labeled sources in training data is small (S1) than the full sources of labeled data (S9). The reason for differences could be explained as EmbNum+ adopt the augmentation technique to learn a better generalize similarity between numerical attributes. It means that EmbNum+ could be useful in practice when we do not have many samples in the labeled dataset.

The performance of semantic labeling systems is different according to datasets. In particular, semantic labeling on City Data, DBpedia NKB, and Wikidata NKB yields higher performance than Open Data. The performance differences occur because of data quality. The City Data, DBpedia NKB, and Wikidata NKB are synthesis data, where each numerical values of attributes are normalized in terms of data scaling. Open Data is the real-world data where we usually do not know the meaning of attributes, therefore it is difficult to do normalization operation. It means that Open Data is more difficult than the other three datasets. Despite the differences between datasets, EmbNum+ results consistently outperform other approaches.

Although, EmbNum+ was trained on 50% of DBpedia NKB, the performance of EmbNum+ consistent yield the best performance in the four datasets, especially the two different datasets: Wikidata NKB, and Open Data. It means that EmbNum+ is promising for semantic labeling in a wide range of data domains.

To understand whether EmbNum+ does significantly outperform Semantic Typer and DSL, we performed a paired sample t-test on the MRR scores between EmbNum+ and Semantic Typer, DSL. Table 4.7 show the result of the paired t-test on City Data, Open Data, DBpedia NKB, and Wikidata NKB. We set the cutoff value for determining statistical significance to 0.01. The results of the paired t-test revealed that EmbNum+ significantly outperforms Semantic Typer and DSL on all four datasets (all the $p\_values < 0.01$).

Semantic Labeling: Efficiency

In this experiment, we also used the same setting with the previous experiment but the efficiency is evaluated by the run-time of semantic labeling. Table 4.8 and Figure 4.13 depict the run-time of semantic labeling of Semantic Typer, DSL, and EmbNum+ on the four dataset: DBpedia NKB, Wikidata NKB, City Data, and Open Data.

Table 4.7: Paried sample t-test between EmbNum+ and Semantic Typer, DSL on DBpedia NKB, City Data, Wikidata NKB, and Open Data

Dataset	Semantic Typer	DSL
DBpedia NKB	3.46e−6	1.40e−4
City Data	2.21e−5	5.99e−5
Wikidata NKB	1.31e−5	2.25e−4
Open Data	9.10e−9	4.27e−8

The run-time of semantic labeling linearly increases with the number of labeled sources. The run-time of DSL was extremely high when the number of labeled data sources increased because three similarity metrics were needed to be calculated. The run-time of Semantic Typer was less than DSL because it only used the KS test as a similarity metric. Semantic labeling with EmbNum+ is significantly faster than Semantic Typer (about 17 times), and DSL (about 46 times). EmbNum+ outperforms the baseline approaches in run-time since the similarity metric of EmbNum+ was calculated directly on extracted feature vectors instead of all original values.

Unseen Semantic Labeling

In this section, we report the experiment results of unseen semantic labeling. Table 4.9 and Figure 4.14 report the MRR score of EmbNum+ when using (EmbNum+) and not using (EmbNum+ NonRe) relevance model. When the number of unseen semantic labels increases, the performance of semantic labeling decreases if we do not use the relevance model. When we use the relevance model, the performance of EmbNum+ changes considerably.

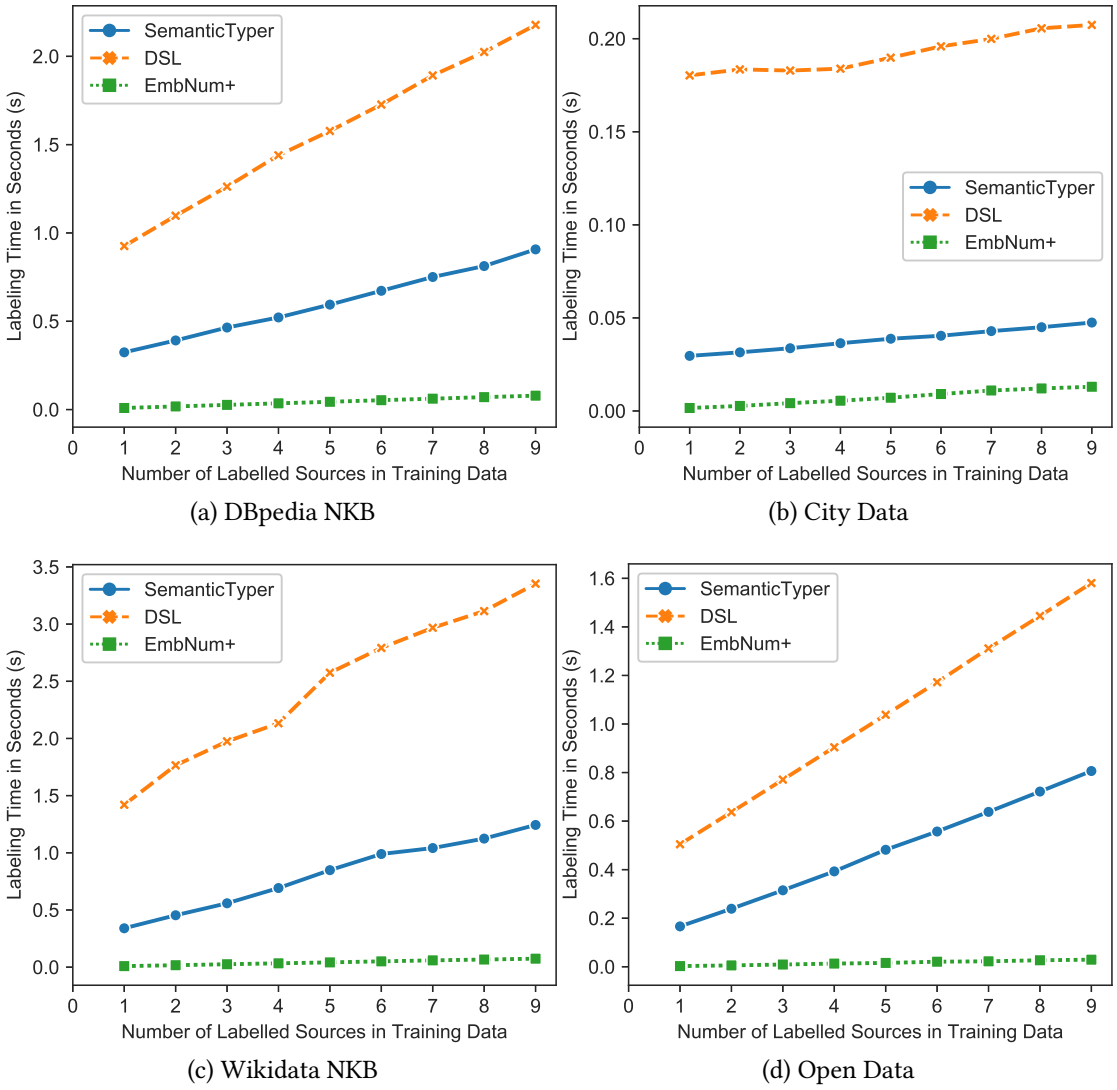
Interestingly, the trend of MRR score changed from decreasing to increasing at 80 % unseen semantic labels in knowledge bases. This result is promising in practice since we usually do not have so many labeled data. Detecting unseen semantic labels assists domain experts in terms of simplifying and reducing the time for the manually labeling process.

Table 4.8: Semantic labeling results of run-time in seconds on DBpedia NKB, City Data, Wikidata NKB, and Open Data

Datasets	Methods	Number of Sources in Labels Sources in Training Data								
		1	2	3	4	5	6	7	8	9
DBpedia NKB	Semantic Typer [2]	0.3241	0.3914	0.4645	0.5216	0.5942	0.6725	0.7506	0.8125	0.9069
	DSL [36]	0.9259	1.0972	1.2621	1.4398	1.5774	1.7269	1.8918	2.0245	2.1777
	EmbNum+ [7]	**0.009**	**0.0178**	**0.0265**	**0.0354**	**0.044**	**0.0531**	**0.0617**	**0.0704**	**0.0787**
City Data	Semantic Typer [2]	0.0296	0.0315	0.0337	0.0364	0.0388	0.0404	0.0429	0.045	0.0475
	DSL [36]	0.1803	0.1835	0.1829	0.1839	0.1898	0.1959	0.1999	0.2056	0.2074
	EmbNum+ [7]	**0.0016**	**0.0027**	**0.0042**	**0.0055**	**0.0071**	**0.0091**	**0.011**	**0.0121**	**0.013**
Wikidata NKB	Semantic Typer [2]	0.34	0.4539	0.5587	0.6919	0.8483	0.9895	1.0416	1.1243	1.2437
	DSL	1.4199	1.7652	1.9741	2.1329	2.5746	2.7915	2.9679	3.1139	3.3529
	EmbNum+ [7]	**0.0091**	**0.0171**	**0.0256**	**0.0336**	**0.0417**	**0.0508**	**0.0595**	**0.0667**	**0.0741**
Open Data	Semantic Typer [2]	0.1663	0.239	0.315	0.3929	0.4819	0.5571	0.638	0.7217	0.8063
	DSL [36]	0.5047	0.637	0.7708	0.9042	1.0378	1.1723	1.3113	1.445	1.5806
	EmbNum+ [7]	**0.0028**	**0.0056**	**0.0092**	**0.0131**	**0.0158**	**0.0208**	**0.0225**	**0.0268**	**0.0292**

Table 4.9: Semantic labeling results of unseen setting in the MRR score on DBpedia NKB, City Data, Wikidata NKB, and Open Data

Dataset	Methods	0%	10%	20%	30%	40%	50%	60%	70%	80%	90%
DBpedia NKB	EmbNum+ NonRe	0.9483	0.8615	0.7679	0.6747	0.5774	0.4869	0.3935	0.2956	0.1977	0.1021
	EmbNum+	0.9483	0.8649	0.7748	0.688	0.5967	0.5179	0.4649	0.4079	0.411	0.4765
City Data	EmbNum+ NonRe	0.9518	0.8674	0.7707	0.6725	0.5696	0.4809	0.3944	0.2983	0.1983	0.1
	EmbNum+	0.9518	0.8774	0.8107	0.7225	0.6762	0.6409	0.6278	0.6217	0.645	0.7867
Wikidata NKB	EmbNum+ NonRe	0.9175	0.8383	0.749	0.6599	0.5621	0.4793	0.3831	0.2913	0.1972	0.0982
	EmbNum+	0.9175	0.8401	0.7573	0.6723	0.5792	0.5095	0.4293	0.383	0.3807	0.4905
Open Data	EmbNum+ NonRe	0.7093	0.6458	0.5777	0.5277	0.4524	0.3811	0.3209	0.2559	0.1663	0.0882
	EmbNum+	0.7093	0.6478	0.5977	0.5557	0.4844	0.4431	0.4169	0.4379	0.4063	0.5202



(a) DBpedia NKB

(b) City Data

(c) Wikidata NKB

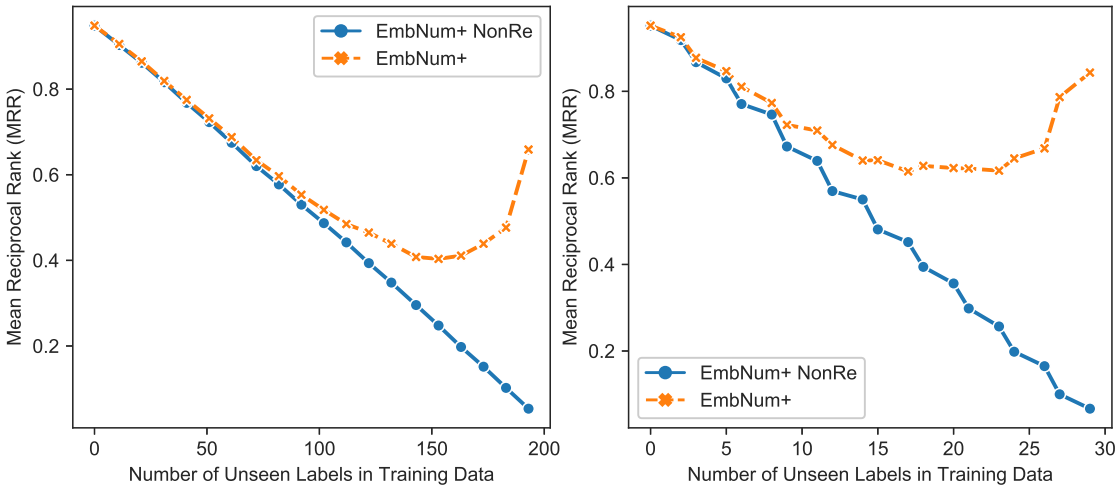
(d) Open Data

Figure 4.13: Semantic labeling results of run-time in seconds on DBpedia NKB, City Data, Wikidata NKB, and Open Data

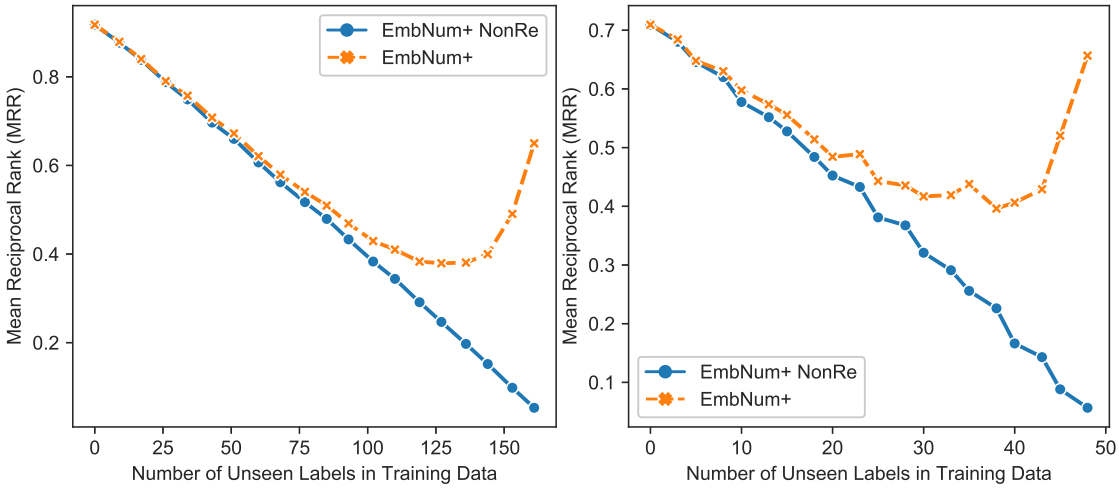
4.4.5 Ablation study

Table 4.10 reports the ablation study of EmbNum+ on City Data, Open Data, DBpedia NKB, and Wikidata NKB.

The method of Num_l1, Num_l2, $Num\_l\infty$ are EmbNum+ without using the representation learning. Among these methods of EmbNum+ without using representation learning, Num_l1 outperforms Num_l2, $Num\_l\infty$. It indicates that the Manhattan



(a) DBpedia NKB (b) City Data



(c) Wikidata NKB (d) Open Data

Figure 4.14: Semantic labeling results of unseen setting in the MRR score on DBpedia NKB, City Data, Wikidata NKB, and Open Data

Table 4.10: Ablation study result of EmbNum+ on City Data, Open Data, DBpedia NKB, and Wikidata NKB

Methods	City Data	Open Data	DBpedia NKB	Wikidata NKB
Num_l1	0.9320	0.6972	0.9026	0.8934
Num_l2	0.9225	0.6891	0.8964	0.8919
$Num\_l\infty$	0.9090	0.6721	0.8879	0.8985
EmbNum+ NonAu	0.9412	0.6998	0.9165	0.9085
EmbNum+	**0.9518**	**0.7093**	**0.9483**	**0.9175**

distance has more advantages than Euclidean and Chevshcesh distance. By removing the representation learning, we can see the MRR score is significantly reduced in the four datasets. This validates our assumption that the representation leaning is a necessary module in the semantic labeling procedure.

EmbNum+ NonAu is a system of EmbNum+ without using the attribute-augmentation module. The performance of EmbNum+ higher than the EmbNum+ NonAu, therefore we verify that the module of attribute-augmentation is necessary for our proposed approach.

4.5 Conclusion

In this section, we introduces EmbNum+, a semantic labeling method that annotates semantic labels for numerical attributes of tabular data. EmbNum+ advances other baseline approaches in many ways. First, EmbNum+ learns directly from numerical attributes without making any assumption regarding data, therefore it is particularly useful to apply on general data when we do not have any knowledge about data domain and distribution. Second, EmbNum+ significantly boosts the efficiency against other baseline approaches because all the calculations are performed on the representation of numerical attributes. Third, EmbNum+ can recognize the unseen query which is extremely helpful in practice. Additional, we also introduced two new synthesis datasets i.e., Wikidata NKB and DBpedia NKB, and one real-world dataset i.e., Open Data. These datasets will be useful to evaluate the scalability as well as the robustness of semantic labeling for numerical values.

In future work, we plan to extend our work in the two directions. The first direction is to extend the similarity metric to interpret multiple scales. In this study, we assumed that two numerical attributes are similar when they are expressed on the same scale. In fact, two attributes with the same meaning could be measured using different scales. For instance, the numerical attributes “human height" could be expressed in “centimeters" or “feet." The second direction is to extend the metric to interpret the hierarchical representation of numerical data. The current presentation using the Euclidean distance cannot reflect the hierarchical structure. Building a similarity metric that is hierarchy-aware can help to make a more fine-grained semantic labeling.

5 MTab: Semantic Annotation for Tabular Data

In Section 5.1, we describe the task of semantic annotation for tabular and information about the challenges. Section 5.1.1 provides definitions as well as notation for the three tasks. We make some assumptions about MTab system in Section 5.1.2. The details of MTab are described in Section 5.2. The evaluation setting, results, and analysis of MTab is reported in Section 5.3. The detail description of other participant are summarized in Section 5.4. Finally, we conclude MTab in Section 5.5 including discussion on limitation, and future work.

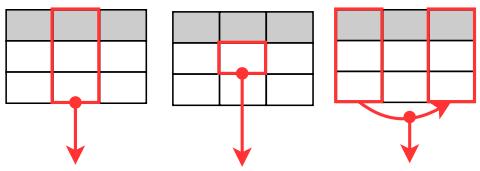
5.1 Introduction

Tabular data annotation is the task of labeling table elements into standard concepts to gain a semantic understanding of data. As a result, such annotation could be useful for other downstream applications such as data integration, learning, mining, machine

learning, and knowledge management and discovery. The annotation tasks are difficult due to the heterogeneous of tabular data. The use of vocabulary and schema are different in different data sources, moreover, these data usually missing, or incomplete metadata or ambiguous, non-standard vocabulary.

A possible solution is to match table elements into the general knowledge graph such as DBpedia, Wikidata, YAGO. The semantic annotation could enhance a wider range of applications such as information retrieval, knowledge management, and construction. Due to the differences in benchmark settings, datasets as well as target matching knowledge bases in the literature, there is a need for a general benchmark for tabular data matching tasks to promote a comparison of annotation systems.

Tabular Data to Knowledge Graph Matching (SemTab 2019)[1] is a challenge on matching table elements into knowledge graph (KG) concepts, especially DBpedia. Fig. 5.1 depicts the three sub-tasks for SemTab 2019. Given a table data, **CTA** (Fig. 5.1a) is the task of assigning a semantic type (e.g., a DBpedia class) to a column. In **CEA** (Fig. 5.1b), a cell is linked to an entity in KG. The relation between two columns is assigned to a property or predicate in KG in **CPA** (Fig. 5.1c).



(a) **CTA** (b) **CEA** (c) **CPA**

Figure 5.1: Tabular Data Matching to Knowledge Graph (DBpedia)

Figure 5.2 depict an example of semantic annotation for tabular data. dbr:Authur_Drews is an entity annotation table cell "A. Drews". The type annotations for the column "col1" are dbo:PopulatedPlace, dbo:Place. dbo:deadYear is the annotation for the relation between column "col0" and column "col2".

The SemTab 2019 challenge contains four round, each round came with a different dataset for each annotation task. In detail, round 1 data is extracted from the T2Dv2 dataset, round 2 is a combination of Wikipedia tables and automatically generated tables from DBpedia, round 3, and round 4 datasets also were automatically generated

[1]SemTab 2019 link: http://www.cs.ox.ac.uk/isg/challenges/sem-tab/

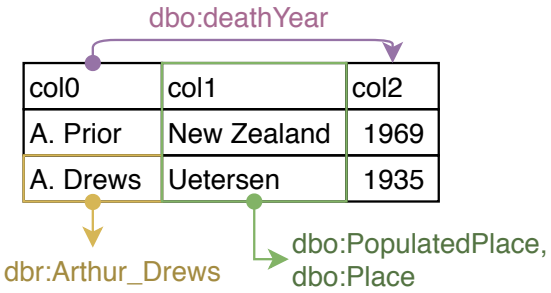


Figure 5.2: Example of semantic annotation for tabular data

from DBpedia. The challenge attached a lot of attention from many research teams, we had seven stable systems across the four-round and annotation tasks. Figure 5.1 depicts number of participants in SemTab 2019.

Table 5.1: Number of participants in SemTab 2019

Number of	**Round 1**	**Round 2**	**Round 3**	**Round 4**
Paticipants	17	11	9	8
CTA	13	9	8	7
CEA	11	10	8	8
CPA	5	7	7	7

We introduce MTab which is a general framework to address all the three tasks of SemTab 2019. MTab combines the voting algorithm and the probability models to solve critical problems of the matching tasks. The results of MTab got the first prize for Semantic Web Challenge on Tabular Data to Knowledge Graph Matching (SemTab 2019).

5.1.1 Problem Definitions

We denotes DBpedia as a knowledge graph $G = (E, T, R)$, where E, T, R are the set of entities, the set of types (or classes), and the set of relations (or predicates) respectively. e is an entity $e \in E$, t_e is the type of an entity $t_e \in T$, and r is a relation of entity-entity or entity-literal $r \in R$.

Let a table S be a two-dimensional tabular structure consisting of an ordered set

of N rows, and M columns. n_i is a row of table ($i = 1...N$), m_j is a column of table ($j = 1...M$). The intersection between a row n_i and a column m_j is $c_{i,j}$ is a value of the cell $S_{i,j}$.

The tabular to KG matching problems could be formalized the three sub-tasks as follows.

- **CEA**: matching a cell value $c_{i,j}$ into a relevance entity $e \in E$.

$$c_{i,j} \xrightarrow{\text{CEA}} E \tag{5.1}$$

- **CTA**: matching a column m_j into a exact relevance type and its ancestors.

$$m_j \xrightarrow{\text{CTA}} T \tag{5.2}$$

- **CPA**: matching the relation between two columns m_{j_1} and m_{j_2} ($j_1, j_2 \in [1, M]$, $j_1 \neq j_2$) into a relation $r \in R$.

$$r_{m_{j_1}, m_{j_2}} \xrightarrow{\text{CPA}} R \tag{5.3}$$

5.1.2 Assumptions

In MTab, we adopt the following assumptions:

Assumption 1 *MTab is built on a closed-world assumption. It means that the target KG is completed and corrected.*

Assumption 2 *The type of input table is vertical relational table.*

Assumption 3 *Input tables are independence, it means there is no sharing information between input tables.*

Assumption 4 *Column cell values have the same entity types and data types.*

Assumption 5 *The first row of table (n_1) is table header. The first cell of column is header of this column, $c_{1,j} \in m_j$.*

In practice, table headers could have more complicated structures. Headers could available or non-available, be located at the beginning of the table or not, have one rows or multiple rows. In this work, we omit those issues and assume that the table header is located at the first row.

5.2 MTab Approach

MTab is built on the joint probability distribution of many signals inspired by the graphical probability model-based approach [16] and the signal (or confidence) propagation fashion between table elements of T2K [13]. However, MTab improves the matching performance by solving two major problems:

- Entity lookup: We found that using only DBpedia lookup (the most popular service) does not usually get relevance entities for non-English queries. Therefore, we perform entity lookup on multiple services (with language parameter) to increase the possibility of finding the relevance entities.

- Literal matching: We found that mapping cell values to corresponding values in a KG are less effective because the corresponding value in KG is rarely equal with a query value. Therefore, with Assumption 4, we adopt literal columns matching to find a relevance relation (property) and aggregate these signals to enhance MTab performance.

Additionally, we also adopt more signals from table elements, introduce a scoring function to estimate the uncertainly from ranking. Note that, the probabilities in this section could be interpreted as subjective confidences.

5.2.1 Framework

We design our system (MTab) as 7-steps pipeline (Fig. 5.3). Step 1 is to pre-process a table data S by decoding textual data, predicting languages, data type, entity type prediction, and entity lookup. Step 2 is to estimate entity candidates for each cell. Step 3 is to estimate type candidates for columns. Step 4 is to estimate relationship between two columns. Step 5 is to re-estimate entity candidates with confidence aggregation

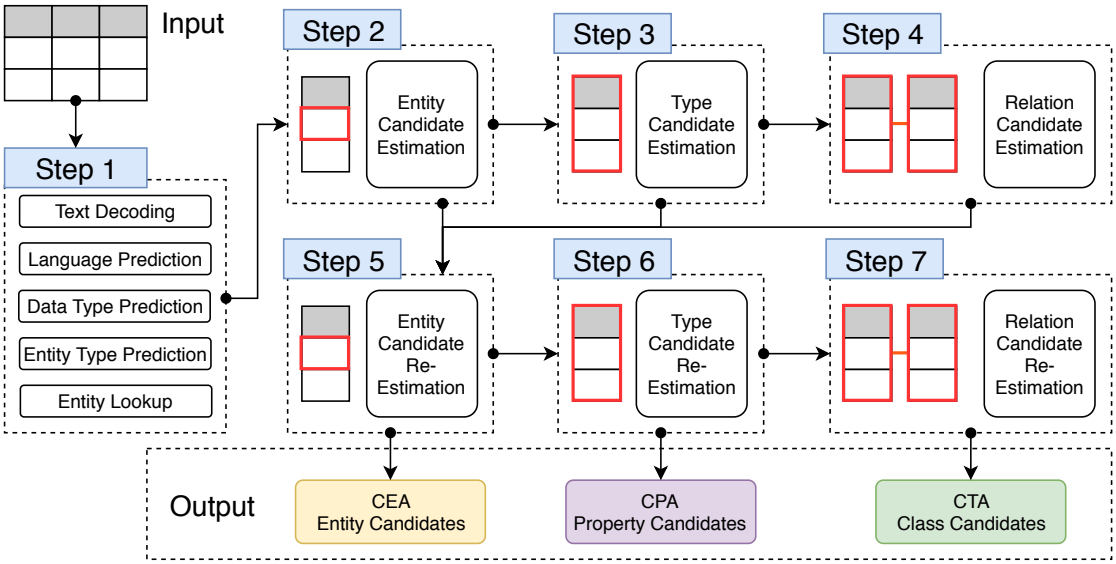


Figure 5.3: MTab framework for tabular data matching

from step 2, step 3, and step 4. Step 6, and Step 7 are to re-estimate type and relation candidates with results from Step 5, respectively.

The following are the detail explanations on each step of the framework.

5.2.2 Step 1: Pre-processing

We perform five processes as follows.

- **Text Decoding:** Reading table data has a problem of textual encoding where some characters are loaded as noisy sequences. Loading incorrect encoding might strongly affect the lookup performance, therefore, we used the ftfy tool [73] to fix all noisy textual data in tables.

- **Language Prediction:** We used the pre-trained fasttext models (126 MB) [74] to predict the languages for tables (concatenate all table cell values) and each cell in the table. Since table data is not only written in English but also in other languages, determining the language of the input query helpful for the lookup tasks.

- **Data Type Prediction:** Next, we perform data type prediction to predict 13

pre-defined data types of duckling [2] for each cell value in a table $c_{i,j}$. Those types are about numerical tags, email, URL, or phone number. If there is no tag assigned, we assign this cell type as a text tag.

- **Entity Type Prediction:** For each cell value in a table $c_{i,j}$, we also perform entity type prediction with the pre-trained SpaCy models [75] (OntoNotes 5 dataset) to predict 18 entity types. If there is no tag assigned, this cell type is assigned to a text tag. We also manually map from those textual entity types (11 entity types) OntoNotes 5 to some DBpedia classes.

- **Entity Lookup:** We search the relevance entity on many services including DBpedia Lookup[3], DBpedia endpoint[4]. Also, we search relevant entities on Wikipedia and Wikidata by redirected links to DBpedia to increase the possibility of finding the relevant entities. We use the language information of the table and cell values as the lookup parameters. If there is any non-English lookup URL, it is redirected to the corresponding English URL. We use α [5] as the limit of lookup results. The search query could be each cell value in a table $c_{i,j}$, or other neighbor cells in the same rows i.

5.2.3 Step 2: Entity Candidate Estimation

In this section, we explain how we estimate the entity candidates. Given a cell value $c_{i,j}$, we have a set of ranking result lists from lookup services $Q_{c_{i,j}}$. q is a list of ranking of entities ordered by degree of relevance of a lookup service, where $q \in Q_{c_{i,j}}$. In MTab, we adopted the four services as DBpedia lookup, DBpedia Endpoint, Wikidata lookup, and Wikipedia lookup. However, we can use any services as long as their output is a ranking list of relevance entities.

Denote $E_{Q_{c_{i,j}}}$ is a set of relevance entities in $Q_{c_{i,j}}$, s_e^q is a confidence score of an entity e where $e \in E_{Q_{c_{i,j}}}$. The confidence score of entity e is calculated as

$$s_e^Q = max(s_e^q) \tag{5.4}$$

[2]Duckling link: https://github.com/facebook/duckling
[3]DBpedia Lookup, link: https://wiki.dbpedia.org/Lookup
[4]DBpedia Endpoint link: https://dbpedia.org/sparql
[5]In MTab, we set $\alpha = 100$

s_e^q is the confidence score of entity e in q.

$$s_e^q = \alpha - rank_e \tag{5.5}$$

where $rank_e$ is the ranking index of entity in q. We normalize those entity confidence score to $[0, 1]$, where $Pr(E_{Q_{c_{i,j}}}|Q_{c_{i,j}}) = 1$,

$$Pr(e|Q_{c_{i,j}}) = \frac{s_e^Q}{\sum\limits_{e \in E_{Q_{c_{i,j}}}} s_e^Q} \tag{5.6}$$

and associate those scores as the potential probability of entities given lookup results.

5.2.4 Step 3: Type Candidate Estimation

Regarding Assumption 4, we categorize table columns to entity columns and literal columns. We aggregate the data type (Duckling Tags and SpaCy Tags) from each cell in a column using majority voting. If the majority tag is text or entity-related, the columns is an entity column, else a numerical column. Regarding numerical columns, we perform semantic labeling with EmbNum+ method [7] to annotate relations (DBpedia properties) for numerical columns [6]. Then, we infer types (DBpedia classes) from those relations.

Numerical Column

The set of numerical columns in table S is M_{num}. Given a numerical column m_j, we use re-trained EmbNum+ model on DBpedia [7] to derive embedding vector for the list of all numerical values of the column and then search the corresponding relations from the database of labeled attributes [7]. The result q_{m_j} is a ranking of relevance numerical attributes in terms of distribution similarity. We also use α as the limit for ranking result. The confidence score of a relation r is calculated as the following equation.

$$s_r^{m_j} = \alpha - rank_r \tag{5.7}$$

[6] We only use EmbNum+ for those columns have at least 10 numerical values
[7] We used all numerical attributes of DBpedia as the labeled data

where $rank_r$ is the ranking index of r. These scores are also normalized to a range of [0,1] to associate the probability of potential of relation for numerical columns $Pr(r|m_j)$.

Next, we use DBpedia endpoint to infer the classes (types) from those relations as Figure 5.4. $T_{q_{m_j}}$ is a set of inferred types, t denotes a type $t \in T_{q_{m_j}}$. Those types will be used for entity columns. The confidence score of types is estimated as the following equation.

$$s_{r_t} = max(s_r^{M_{num}}) \tag{5.8}$$

Then, we normalized those scores to [0,1] so that $Pr(T_{q_{M_{num}}}) = 1$, those confidence scores are associated as the probabilities of type potential $Pr(t|M_{num})$ given M_{num}.

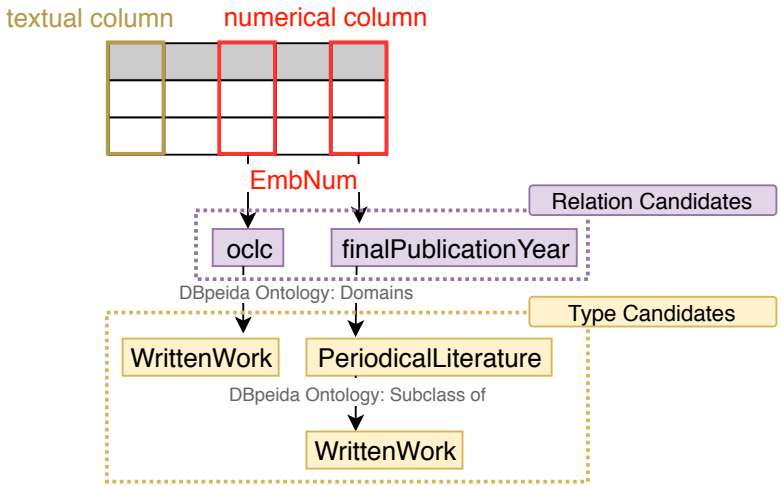


Figure 5.4: Property lookup with EmbNum

Entity Column

Given a set of entity columns in table S is M_{ent}, we consider these signals from

1. $Pr(t|M_{num})$: the probabilities of type potential from numerical columns
2. $Pr(t|m_j, Q_{m_j})$: the probabilities of type potential aggregated from the types of entity lookup for the all cells in column m_j.

$$Pr(t|m_j, Q_{m_j}) = \sum_{c_{i,j} \in m_j} Pr(t|Q_{c_{i,j}}) \tag{5.9}$$

We normalized these aggregated potentials and associates these as potential probabilities.

3. $Pr(t|m_j, SpaCy_{m_j})$: the probabilities of type potential aggregated from SpaCy entity type prediction for the all cell in column m_j. We used majority voting and normalized these voting values to [0,1]. Then, we associate those normalized voting value type potential probabilities.

4. $Pr(t|c_{1,j})$: the probabilities of type potential given header value of the column m_j. We associate the normalized Levenshtein distance as potential probability that a type (DBpedia class) correspond with a header value.

The probabilities of type potential is derived from the four signals as the following equation.

$$Pr(t|m_j) = w_1 Pr(t|M_{num}) w_2 Pr(t|m_j, Q_{m_j}) w_3 Pr(t|m_j, SpaCy_{m_j}) w_4 Pr(t|c_{1,j}) \quad (5.10)$$

where w_1, w_2, w_3, w_4 are learnable weights. Note that, some probabilities of signals might be 0 or too small, and aggregate those might add too much noise to the final aggregation. Therefore, if any signal probabilities less than β[8], we omit those signals. After aggregation, we also perform normalization for $Pr(t|m_j)$ to a range of [0,1] so that $Pr(T_{m_j}|m_j) = 1$.

5.2.5 Step 4: Relation Candidate Estimation

Given two columns m_{j_1} and m_{j_2}, we estimate the probabilities of relation potential of $Pr(r|m_{j_1}, m_{j_2})$. We consider two type of relation between two columns: Entity column to Entity column and Entity column to non-Entity column. To be simple, we associate the first entity column is m_{j_1}. If the second column is entity column, we denote it as $m^{ent}_{j_2}$, else $m^{non-ent}_{j_2}$.

Entity - Entity columns $Pr(r|m_{j_1}, m^{ent}_{j_2})$:

Given c_{i,j_1} is a cell value of the column m_{j_1} and the row r_i, c_{i,j_2} is a cell value of the column $m^{ent}_{j_2}$. We assume that there is a relation between entity candidates of c_{i,j_1}

[8] In MTab, $\beta = 0.5$

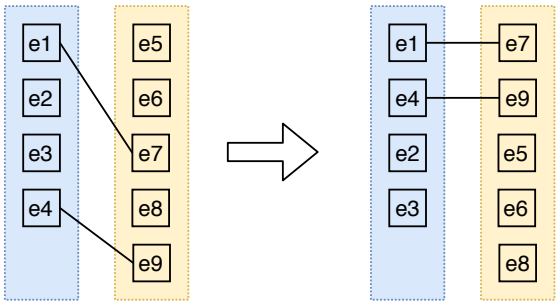


Figure 5.5: Illustration of entity candidate re-ranking between two column cells

and c_{i,j_2}, therefore we use DBpedia endpoint to find how many links (relations or properties) between entity candidates of c_{i,j_1} and c_{i,j_2}. The confidence score of relation is calculated as the following equation. $s_r^{c_{i,j_1},c_{i,j_2}} = 1$ if there is any relation between entity candidates of two columns. Then, we aggregate those scores of all rows to get the candidate score for two columns as the following equation.

$$s_r^{m_{j_1},m_{j_2}^{ent}} = \sum_{i \in [1,N]} s_r^{c_{i,j_1},c_{i,j_2}} \tag{5.11}$$

Then, we normalize those score to a range of [0,1] so that $Pr(R_{m_{j_1},m_{j_2}^{ent}}|m_{j_1}, m_{j_2}^{ent}) = 1$ and associate it as the probability of relation potential of Entity and Entity Columns $Pr(r|m_{j_1}, m_{j_2}^{ent})$.

Figure 5.5 illustrate the re-ranking between the two entity candidates from two-column cells in the same row.

Entity - Non-Entity columns $Pr(r|m_{j_1}, m_{j_2}^{non-ent})$:

Given c_{i,j_1} is a cell value of the column m_{j_1} and the row r_i, c_{i,j_2} is a cell value of the column $m_{j_2}^{non-ent}$. We estimate the relevance ratio between entity candidates and non-entity value c_{i,j_2}. Given an entity candidate e have pairs of relation(r_e)-values(v_e), we compare the non-entity value c_{i,j_2} with all attribute values v_e. We select those pairs have ratio larger than β. We only compare two values of c_{i,j_1} and v_e based on there data types (textual type or numerical type).

- For textual values: We use the normalized Levenshtein distance to estimate the relevance ratio between v_e and c_{i,j_2} as $s(v_e, c_{i,j_2})$.

- For numerical values: the relevance ratio is calculated as the following equation.

$$s(v_e, c_{i,j_2}) = \begin{cases} 0, & \text{if } max(|c_{i,j_2}|, |v_e|) = 0 \text{ and } |c_{i,j_2} - v_e| \neq 0 \\ 1, & \text{if } max(|c_{i,j_2}|, |v_e|) = 0 \text{ and } |c_{i,j_2} - v_e| = 0 \\ 1 - \frac{|c_{i,j_2} - v_e|}{max(|c_{i,j_2}|, |v_e|)}, & \text{if } max(|c_{i,j_2}|, |v_e|) \neq 0 \end{cases} \tag{5.12}$$

We aggregate all relevance ratio with respect to relations. Then we normalize those aggregated ratio to [0,1], and associate this as probability of relation potential. $Pr(r|m_{j_1}, m_{j_2})$. If the column of m_{j_2} is numerical columns, we also aggregate the re-calculated probability from $Pr(r|m_{j_2})$ (step 3) as the following equation.

$$Pr(r|m_{j_1}, m_{j_2}, m_{j_2}\text{is numerical}) = w_5 Pr(r|m_{j_1}, m_{j_2}) w_6 Pr(r|m_{j_2}) \tag{5.13}$$

where w_5, w_6 are learnable parameters.

5.2.6 Step 5: Entity candidate Re-Estimation

In this step, we present a method to re-estimate the probabilities of entity candidates $Pr(e|S)$. Given a cell $S_{i,j}$ containing a cell value $c_{i,j}$ at row n_i, and column m_j, we consider these signals from:

- $Pr(e|Q_{c_{i,j}})$: The entity candidate probabilities given look up results.
- $Pr(e|m_j)$: The probabilities of entity candidates given their type's probabilities (Step 3). This can be estimated as the following equation.

$$Pr(e|m_j) = max(Pr(t_e|m_j, Q_{m_j})) \tag{5.14}$$

  where t_e is a type of the entity e.

- $Pr(e|c_{i,j})$: The probabilities of entity candidates given the cell value $c_{i,j}$. We get the mean ratio of the normalized Levenshtein distance, heuristic abbreviation rules (first character of words, titles, dates, time).
- $Pr(e|n_i, m_{j_1})$: The probabilities of entity candidates given cell values in a row $c_{i,j} \in n_i$. We do the same procedure as Step 4 to compare all entity values with a

cell value, and compute the mean probability for all cell value in a row as the following equation.

$$Pr(e|n_i, m_{j_1}) = mean(Pr(e|m_{j_1}, m_{j_2})) \tag{5.15}$$

where $j_1 \neq j_2$.

Overall, the equation is as follows.

$$Pr(e|S) = w_7 Pr(e|Q_{c_{i,j}}) w_8 Pr(e|m_j) w_9 Pr(e|c_{i,j}) w_{10} Pr(e|n_i, m_{j_1}) \tag{5.16}$$

where w_7, w_8, w_9, w_{10} are learnable parameters.

5.2.7 Step 6, 7: Re-Estimate Types and Relations

We select the highest probabilities of entity candidates in Step 5 for each cell $S_{i,j}$ to re-estimate types and relations with majority voting.

5.3 Evaluation

In this section, we first report the detail about benchmark datasets in Section 5.3.1, evaluation metrics in Section 5.3.2. The overall results are reported in Section 5.3.3. The detail error analysis, and improvement are described in Section 5.3.4. Finally, we provide detail an ablation study on the contribution of EmbNum+ in MTab system (Section 5.3.5).

5.3.1 Benchmark Datasets

The SemTab 2019 challenge contains four rounds, each round came with a different set of tables as well as the targets of matching for each annotation task. In detail, round 1 data is extracted from the T2Dv2 dataset, round 2 is a combination of Wikipedia tables and automatically generated tables from DBpedia, round 3, and round 4 datasets also were automatically generated from DBpedia. To generate the tabular data, first a list of classes and properties are gathered, then for each class, the generator selects groups of properties and using them to create "realistic" tables using SPARQL queries. Finally,

these "realistic" tables were added noise into the surface textual of table cells or remove "easy" matches cells.

Table 5.2 reports the statistic about the SemTab 2019 dataset [76]. Round 1 dataset was extracted from the T2Dv2 dataset which is a standard dataset in tabular data annotation. Round 2 dataset is the biggest and most complex one since it was combined from two different datasets of Wikipedia tables and DBpedia generated tables. Round 3 and Round 4 are generated tables, but in Round 4, the easily matched cells were removed.

Table 5.2: SemTab 2019 dataset

	# Table	# Target CEA	# Target CTA	# Target CPA
Round 1	70	8,418	120	116
Round 2	11,925	463,773	14,561	6,762
Round 3	2,162	406,827	5,762	7,575
Round 4	818	107,352	1,732	2,747

5.3.2 Evaluation Metrics

There are four different metrics used to evaluate tabular data annotation:

F1-score is a harmonic mean of precision and recall. It is used as the primary score to measure the the performance of entity annotations (**CEA** - all rounds), relation annotations (**CPA** - all rounds), and type annotation (**CTA** - round 1). The F1 metric is calculated as follows.

$$\text{F1} = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \tag{5.17}$$

where Precision, and Recall are calculated as follows.

$$\text{Precision} = \frac{\text{\# correct annotations}}{\text{\# annotations}} \tag{5.18}$$

$$\text{Recall} = \frac{\text{\# correct annotations}}{\text{\# target annotations}} \tag{5.19}$$

The Precision scores was also used as the secondary score in entity annotations (**CEA** - all rounds), relation annotations (**CPA** - all rounds), and type annotation (**CTA** - round 1).

Regarding the type annotation **CTA** task, there are two metrics designed to measure the hierarchical of class annotations (Average Hierarchical - AH) and perfect class annotations (Average Perfect - AP). The AH score is used as the primary score, while AP score is used as the secondary score for round 2, 3, 4 of **CTA** task.

Denote that the list of target columns is T. A column annotation is denoted as a, and the number of perfect annotations denotes as p_a, the number of OK annotation denotes as o_a, and the number of the wrong annotation denote as w_a. The equations of the AH score and AP score are described as follows.

$$\text{AH} = \frac{\sum_{a \in T} p_a + 0.5 * o_a - w_a}{|T|} \tag{5.20}$$

$$\text{AP} = \frac{\sum_{a \in T} p_a}{\sum_{a \in T} p_a + o_a + w_a} \tag{5.21}$$

5.3.3 Experimental Results

Table 5.3 (CEA[9]), Table 5.4 (CTA[10]), Table 5.5 (CPA [11]) reports the overall results of MTab for three matching tasks in the four rounds of SemTab 2019. Overall, these results show that MTab achieves the best performance performances for all the three matching tasks in all of the rounds.

The MTab performance might be explained in part by tackle the two major problems of the three matching tasks. MTab performed language prediction and lookup with the language parameter. Moreover, MTab built on top of multiple lookup services, therefore, it increases the possibility of finding the relevant entities. Additionally, MTab adopted many new signals (literal) from table elements and use them to enhance

[9] CEA full results: https://www.aicrowd.com/challenges/iswc-2019-cell-entity-annotation-cea-challenge/leaderboards

[10] CTA full results: https://www.aicrowd.com/challenges/iswc-2019-column-type-annotation-cta-challenge/leaderboards

[11] CPA full results: https://www.aicrowd.com/challenges/iswc-2019-columns-property-annotation-cpa-challenge/leaderboards

matching performance.

Table 5.3: Entity annotation results in F1 score for the four rounds of SemTab 2019 (7 stable systems)

CEA	Round 1	Round 2	Round 3	Round 4
MTab	**1.000**	**0.911**	**0.970**	**0.983**
CSV2KG	-	0.883	0.961	0.907
Tabularisi	0.884	0.808	0.751	0.803
MantisTable	1	0.614	0.618	0.973
LOD4ALL	0.852	0.757	0.828	0.648
ADOG	-	0.742	0.911	0.835
DAGOBAH	0.897	0.713	0.689	0.578

Table 5.4: Type annotation results in AH score for the four rounds of SemTab 2019 (7 stable systems)

CTA	Round 1 (F1)	Round 2	Round 3	Round 4
MTab	**1.000**	**1.414**	**1.956**	**2.012**
CSV2KG	0.833	1.376	1.864	1.846
Tabularisi	0.825	1.099	1.702	1.716
MantisTable	0.929	1.049	1.648	1.682
LOD4ALL	0.925	0.893	1.442	1.071
ADOG	0.908	0.713	1.409	1.538
DAGOBAH	0.644	0.641	0.745	0.684

5.3.4 Error Analysis and Improvement

In this section, we analyze the error cases of entity matching (Section 5.3.4), type matching (Section 5.3.4), and relation matching (Section 5.3.4) of MTab system. We report the analysis results on Round 2, 3, 4 dataset due to the larger number of tables,

[11] https://github.com/phucty/MTab

Table 5.5: Relation annotation results in F1 score for the four rounds of SemTab 2019 (7 stable systems)

CPA	Round 1	Round 2	Round 3	Round 4
MTab	**0.987**	**0.881**	**0.844**	**0.832**
CSV2KG	-	0.877	0.841	0.83
Tabularisi	0.606	0.79	0.827	0.823
MantisTable	0.965	0.46	0.518	0.787
LOD4ALL	-	0.555	0.545	0.439
ADOG	-	0.459	0.558	0.75
DAGOBAH	0.415	0.713	0.519	0.398

as well as variety table size [76]. The Round 1 dataset is a subset of a T2Dv2 dataset, this dataset is used as a demo sample data about the annotation tasks.

CEA: Entity Matching

Regarding entity matching ground truth, we found that many samples are inconsistently URI encoded and decoded representation. For example An entity URI of dbr:Angélica_Rivera could be encoded as "dbr:Ang%C3%A9lica_Rivera" and decoded as "dbr:Angélica_Rivera". The ground truth of CEA contains a mixture between encoded URI and decoded URI. According to URI encoding of DBpedia, the encoding URI (percent-encoding) is not encouraged [12]. We verify how many samples do not have decoded URI in CEA ground truth in Table 5.6. Note that, we only focus on the large dataset such as Round 2, 3, 4.

Table 5.6: Number of none decoded URI samples in CEA ground truth in SemTab 2019

Ground Truth	# Samples	# No decode samples
Round 2	463,773	12,200 (2.63%)
Round 3	406,827	3,273 (0.8%)
Round 4	107,352	174 (0.16%)

[12]DBpedia URI encoding: https://wiki.dbpedia.org/uri-encoding

To make the URI consistency, we provide the encoded and decoded URI for each URI in the original CEA ground truth. Then, we measure the MTab performance on the new ground truth called EDCEA_GT (Encoded and Decoded CEA ground truth). In Table 5.7, the performance of MTab slightly improve when testing on EDCEA_GT.

Table 5.7: Comparison of MTab performance in the original CEA ground truth (CEA_GT) and the new CEA ground truth (EDCEA_GT)

	CEA_GT	EDCEA_GT
Round 2	0.911	0.916
Round 3	0.970	0.978
Round 4	0.983	0.984

CTA: Type Matching

In this section, we provide a detailed analysis of the evaluation metrics of CTA tasks as well as error analysis on the incorrect annotations of the MTab system.

Evaluation metrics Regarding evaluation metrics of CTA, the AH, and AP scores are difficult to be interpreted because we do not know the maximum and minimum values of those scores (the scoring equations are based on reward 1 point perfect types, 0.5 points for OK types and penalty -1 point for wrong types).

We propose two new scoring functions based on the AH and AP scores, but we normalize these scores to the range of [0,1], so that could be easy to compare between different datasets. Denote that the list of target columns is T. A column annotation is denoted as a, and the number of perfect annotations denotes as p_a, the number of OK annotation denotes as o_a, and the number of the wrong annotation denote as w_a. The new equations of NAH (Normalized Average Hierarchical) score and NAP (Normalized Average Perfect) score are described as follows.

$$NAH = \frac{\sum_{a \in T} max(0, \frac{p_a + o_a - w_a}{p_a + o_a + w_a})}{|T|} \tag{5.22}$$

$$NAP = \frac{\sum_{a \in T} p_a}{|T|} \tag{5.23}$$

Table 5.8 reports the maximum bound of annotations using two old scoring AH, AP, and two normalized scores NAH, and NAP. In this experiment, we use the ground truth to simulate the situation of getting perfect annotations. The maximum score of AH and AP score vary in different datasets, as a result, it is difficult for concrete comparisons, while the normalized scores could do this purpose.

Table 5.8: Comparison between normalized metrics NAH, NAP and the original AH, AP score of CTA tasks on the perfect annotations

CTA	AH	AP	NAH	NAP
Round 2	2.254	0.285	1.000	1.000
Round 3	2.444	0.257	1.000	1.000
Round 4	2.232	0.289	1.000	1.000

Table 5.9 depicts the comparison using MTab between result of MTab in CTA tasks on the old metrics (AH, AP) and new metrics (NAH, NAP) using MTab.

Table 5.9: Comparison between normalized metrics NAH, NAP and the original AH, AP score of CTA tasks using MTab

CTA	AH	AP	NAH	NAP
Round 2	1.412	0.276	0.837	0.767
Round 3	1.956	0.261	0.897	0.924
Round 4	2.012	0.300	0.973	0.963

Error analysis In this section, we report two types of CTA errors of MTab system: incorrect type annotations, and incorrect branch annotations. The incorrect type annotations are those annotations be assigned in different type such as the annotation is [dbo:TelevisionShow, dbo:Work] where the ground truth is [dbo:Person]. The incorrect branch annotations are those annotations that be assigned in the same abstract type but different subtypes such as the annotation are [dbo:Writer, dbo:Person]

where the ground truth is [dbo:Person]. In this case, the annotation of dbo:Writer is incorrect but the abstract type dbo:Person is correct.

Table 5.10: Error analysis of MTab on CTA tasks

Errors	Type	Branch	Total
Round 2	697 (40.96%)	1005 (59.04%)	1702
Round 3	72 (14.94%)	410 (85.06%)	482
Round 4	20 (35.71%)	36 (64.29%)	56

Table 5.10 reports the details statistic of total errors and two error types of MTab for CTA tasks. The incorrect branch annotation has a larger number of errors than the type annotations. The number of error annotations in Round 4 is the smallest one. It could be explained as the majority voting on entity types rely on the CEA entity annotation tasks, the CEA results also got the highest performance in the Round 4 dataset. Improving the entity annotations (CEA) could improve the performance of type annotations (CTA).

CPA: Relation Matching

Regarding relation matching, we found that the ground truth annotations do not have the equivalent relations. For example, the relation of dbo:team has its equivalent relation as dbo:club. The direct equivalent properties in DBpedia endpoint are (dbo:team, dbo:club), (dbo:language, dbo:deFactoLanguage, dbo:jureLanguage), and (dbo:area, dbo:landArea, dbo:waterArea). We add those direct equivalent properties into the ground truth (CPA_GT) and associate the new ground truth as DECPA_GT (Direct Equivalent CPA Ground Truth).

Table 5.11 reports a comparison between MTab performance in the original CPA_GT and the new ground truth DECPA_GT. The performance of MTab on CPA significant improve when tested in DECPA_GT.

Due to the incompleteness of DBpedia, we found that there are other indirect equivalent relations in DBpedia. For example, dbo:deathCause and dbo:causeOfDeath have the same equivalent property of wikidata:P509 (cause of death). The problem of knowledge graph completion is not the main focus of this work, but we can expect the

Table 5.11: Comparison between MTab performance in the original CPA_GT and the new ground truth DECPA_GT

CPA	# Errors		F1 score	
	CPA_GT	DECPA_GT	CPA_GT	DECPA_GT
Round 2	1,090	388	0.839	0.888
Round 3	1,208	939	0.844	0.875
Round 4	457	301	0.833	0.890

improvement of relation annotations when the completeness of DBpedia is improved.

5.3.5 Ablation study: Contributions of EmbNum+ in MTab

In this section, we provide an ablation study to understand the contribution of EmbNum+ in the MTab system. We create model MTab- where the signals from numerical columns (Step 3) do not take consideration in the joint probability of entity columns. Then, we test MTab- on the original ground truth of the three tasks CEA, CTA, and CPA.

Table 5.12: Annotations of MTab- on CEA, CTA, and CPA tasks

MTab-	CEA (F1)	CTA (AH)	CPA (F1)
Round 2	0.891 (-2.1%)	1.271 (-10.1%)	0.831 (-5.7%)
Round 3	0.954 (-1,7%)	1.725 (-11.8%)	0.839 (-0.6%)
Round 4	0.955 (-2.8%)	1.703 (-15.4%)	0.831 (-0.1%)

Table 5.12 report the results of MTab- on the primary scores (F1-score in CEA and CPA tasks, AH score in CTA task). We observe that MTab- where the EmbNum+ does not take into account, achieves a lower performance than MTab. These results validate that EmbNum+ is a necessary module in the MTab system.

5.4 Related Work in SemTab 2019

In this section, we describe the six other systems frequently participants for all rounds of SemTab 2019 challenges.

In general, all of the participants start with generate entity candidates by lookup table cell values or searching those values in local index with Elastic Search in DBpedia, Wikidata. The details about the lookup services are reported in Table 5.13. Then, the type candidates and relation candidates are estimated using the entity candidates. Then, re-estimate the entity candidates with the type and relation candidates.

CSV2KG (IDLAB) first search on DBpedia lookup and DBpedia Spotlight to generate entity candidates [77]. The type candidates and relation annotations are estimated using majority voting approaches based on entity candidates. Then, the entity annotations are estimated using the information of relation candidates. Finally, type annotations are estimated using entity annotations.

Tabular ISI approach first generates entity candidates with Wikidata API, and Elastic Search on entity labels of Wikidata, DBpedia. Second, the authors use the heuristics TF-IDF approach and machine learning (neural network ranking) model to select the best candidate for the entity annotation task [78]. The type annotations are estimated with the results from entity annotations with hierarchy searching on common classes. The relation annotations are estimated by finding the relation between entity candidates of the primary and secondary columns or value matching between the primary entity candidates and the value of the secondary columns.

Mantis Table performs column analysis including predicting name entity columns, literal columns, and subject column, then mapping between columns into concepts in DBpedia [79]. Then the relationships between the main column and other columns are estimated based on predicate context and predicate frequency of column value and candidate predicates. Finally, entity linking is performed using the results from previous steps for cell value disambiguation. The relation annotations are estimated by getting the maximum frequency of relation candidates in the entity linking phase. To estimate type annotations, the authors calculate the hierarchical path score of entity types from entity annotations. Then type annotations are estimated on the maximum of the path score.

DAGOBAH performs entity linking with a lookup on Wikidata and DBpedia as

well as voting mechanisms [80]. The authors used Wikidata entity embedding to estimate the entity type candidates with the assumption that entities in the same column should be closed in the embedding spaces as they share semantic meanings.

LOD4ALL uses a combination of direct search (SPARQL ASK on dbr:"query"), keyword search (Abbreviation of Human name) and Elastic Search to find entity candidates [81]. The entity candidates will be used to estimate type and relation candidates.

ADOG focuses on the task of entity annotation with Elastic Search on an integrated ontology (DBpedia sub-graph) using ArangoDB [82]. The results of type and relation annotations are estimated from entity annotations.

In summary, all these methods focus on lookup services of DBpedia, Wikidata, but such services do not usually return relevance entities for non-English queries. In MTab, we address the problem of non-results entities at the lookup step by predicting the language used in a table, and lookup on multiple services using language parameters. The aggregation on these lookup results increases the possibility to find relevance entities for general tabular data.

Moreover, these tabular data contain many numerical attributes that are helpful if we use the results from semantic labeling for numerical attributes. In MTab, we aggregate signal from the result of semantic labeling for numerical attributes (columns) using EmbNum+ [7] (deep metric for distribution similarity calculation)

5.5 Conclusion

In this chapter, we present MTab for the tabular data matching into Knowledge Base - SemTab 2019. MTab is built on top of multiple lookup services, therefore, it increases the possibility of finding the relevant entities. Additionally, MTab adopted many new signals (literal) from table elements and use them to enhance matching performance.

5.5.1 Limitations

Since MTab is built on the top of lookup services, therefore, the upper bound of accuracy strongly relies on the lookup results. In MTab, it is computation-intensive because of aggregating the confidence signals from many parts of the table. Therefore,

MTab is not suitable for the real-time application, where we need to get the result as fast as possible. MTab could be modified to match only some parts of the table to reduce the processing time as Table Miner+ [21]. However, we find that this is a trade-off between effectiveness and efficiency when using Table Miner+ [21] method. A concrete analysis of the trade-off issue is left as our future investigation.

5.5.2 Future Works

MTab could be improved in many dimensions such as effectiveness, efficiency, and generality. Regarding efficiency, MTab could be modified as parallel processing fashion, since the lookup steps and these probability estimations in Step 2, 3, and 4 are independence. Regarding effectiveness, MTab performance could be improved by relaxing our assumptions:

- Assumption 1: The closed-world assumption might not hold in practice. Improving the completeness and correctness of knowledge graphs might improve MTab performance.
- Assumption 2: Classify table types before matching could help to improve MTab performance [1].
- Assumption 3: In reality, some tables could have shared schema. For example, tables on the Web could be divided into many web pages, therefore we can expect improving matching performance by stitching those tables on the same web page (or domain) [24], [42]. Therefore, performing holistic matching could help improve MTab performance.
- Assumption 5: Correctly recognize table headers could help to improve MTab performance.

Moreover, to evaluate the generality of MTab, it is necessary to test MTab performance on the full version of standard data of tabular data annotation such as T2Dv2 [13], WDC [61], and Wikipedia Tables [29].

[13] T2Dv2 Dataset link: http://webdatacommons.org/webtables/goldstandardV2.html

Table 5.13: Comparison of entity candidate generation methods of SemTab 2019 participants

	MTab	CSV2KG	TabularISI	MantisTable	LOD4ALL	ADOG	DAGOBAH
URI heuristic*	x	✓	x	x	✓	x	x
DBpedia SPARQL	✓	x	x	✓	x	x	✓
DBpedia Lookup	✓	✓	x	x	x	x	x
Dbpeida Spotlight	x	✓	x	x	x	x	x
Wikidata SPARQL	✓	x	✓	x	x	x	✓
Wikipedia (CirrusSearch)	x	x	x	x	x	x	✓
Wikipedia (Multilingual)	✓	x	x	x	x	x	x
DBpedia Elastic Search	x	x	✓	x	x	✓	x
Wikidata Elastic Search	x	x	✓	x	x	x	✓
LOD4ALL Elastic Search	x	x	x	x	✓	x	x

* URI heuristic is to check whether a entity URI exits (dbr:entity_uri) when a table cell value is replaced the space with underscore (dbr:cell_value).

6
Conclusion

This chapter summaries our contributions of this thesis including research impact in Section 6.1, limitation, and future direction in Section 6.2.

6.1 Contributions and Research Impact

The tremendous increasing of tabular data on the Web and Open Data Portals over the past decade creates a huge potential for accessibility, transparency, innovation. Integrating those tabular data into standard knowledge bases enables those data useful for other downstream applications such as information retrieval, knowledge management.

The current approaches on tabular data annotation focus on relational tables, where table cells could be matched into entities in knowledge bases. However, there are a lot of tabular data which do not contain textual content (or contains but can not be matched into knowledge graphs), but they have a large number of numerical values. Many methods ignore those tabular data, as a result, it leads to incorrect propagation in the annotation process. In this thesis, we proposed two methods to address the semantic annotation for numerical values in tables. On the one hand, it could be used in difficult tables where there are no matching entities but contains numerical values (DBS [6], EmbNum+[7]). On the other hand, it also could be used to enhance the annotation performances in general frameworks (MTab [8]).

Common approaches in the semantic annotation for numerical values used p value-based metrics to estimate the similarity between numerical attributes. However, there are several issues (Section 3.1) about p values so that these metrics are not robust in general cases. We first proposed distribution-based metrics DBS [6] to solve these issues of p value-based metrics. The experiments on City Data and Open Data shows that DBS outperform other baselines methods (Semantic Typer [2] and DSL [36]) in a large margin.

EmbNum+ is a deep metric approach learned directly from numerical attributes which are a combination between convolutional neuron network and triplet network [7]. EmbNum+ is built on the top of DBS which learns vector representations for distributions while the similarity metric is calculated directly from two distributions in DBS. The experimental results show that EmbNum+ consistent outperform other bases line approaches. EmbNum+ achieve a higher 1.7%-5.1% performance result than DBS in

the four datasets.

One of the other problems in tabular data matching is that the representation of table cells is complicated since it could be in multiple-language, with strange encoding. Previous approaches performed lookup those table cells directly on knowledge bases (Elastic Search, DBpedia lookup or, Spotlight) then lead to no retrieval results. In this thesis, we proposed the MTab framework solving this problem [8]. We perform language detection on table cells and lookup with language parameters on multiple-lookup services which yield promising performances in entity lookup. Additionally, MTab adopts many new signals (literal) from table elements and uses them to enhance matching performances. Overall, MTab consistently achieves the best performance in the three matching tasks e.g, entity annotation, type annotation, and relation annotation in the Semantic Web Challenge in Tabular Data to Knowledge Graph Matching (SemTab 2019).

6.2 Limitations and Future Works

This section provides the discussions on thesis limitations and the future direction for the tasks of semantic tabular data annotation.

A future direction for semantic annotation for numerical attributes is to increase the metric coverage for multiple scaling of data. Due to the heterogeneous data representation, numerical attributes could be represented in different scaling in different data resources. One possible solution is on increasing the variety in data scaling of numerical knowledge bases. For instance, one numerical attribute of "height in meter" could be augmented into many other numerical attributes in different scalings, such as "height in millimeter", "height in kilometer" and so on. However, the increasing variety and volume of numerical attributes could lead to increasing complexity in similarity metric learning. The details of learning a metric that could be interpreted multiple data scaling will be left as our future work.

One interesting direction is on interpreting the hierarchical semantic representation of numerical attributes. Neumaier et al. [5] introduces a numerical knowledge base structured from DBpedia ontology (classes) with unsupervised clustering. Nickel et al. [83] proposes a method to embed entities into hyperbolic spaces in terms of their type relationships. In EmbNum+, we used Euclidean space to estimate similarity which

is cannot reflect hierarchical structure. Learning a metric could be represented the hierarchical information between numerical attributes is one of our future work.

Regarding the general framework for tabular data matching, the main focus of this thesis is on semantic annotation (MTab), however, the other tasks on table structure annotations also need to be addressed such as table type prediction, table heading prediction, data type parsing, core attribute prediction, and holistic matching. The efficiency of MTab could be improved with parallel processing or embedding techniques, while the MTab effectiveness could be improved with relaxing MTab assumptions. We also plan to evaluate the MTab performance in other domains such as tables in documents, or academic articles.

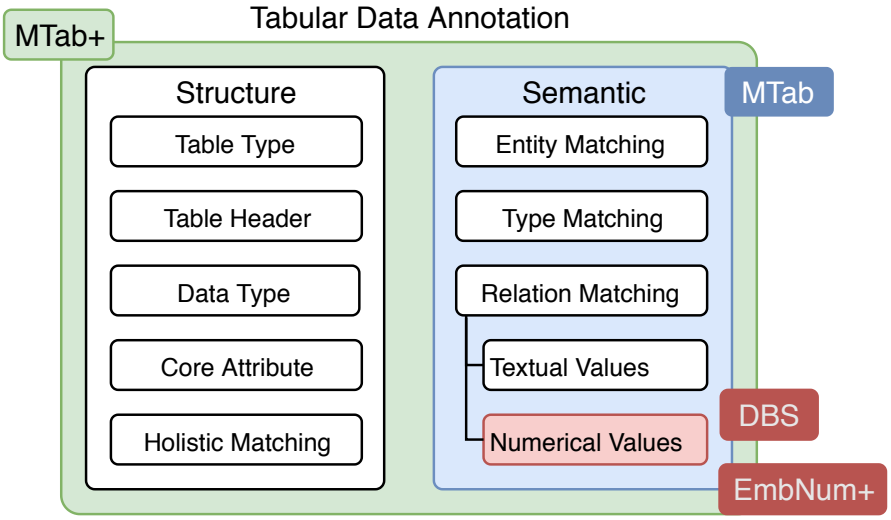


Figure 6.1: MTab+: Tabular Data Annotations

Current approaches on table annotations or interpretations focus on the relational table types, there other table types like matrix types, or entity types are still as open questions, and need to be addressed in the future. MTab achieves very promising performance on the SemTab 2019 dataset, however, it is built on a specific assumption that the input tables are vertical relational tables. The general table data could be represented in different table types, therefore, it is more challenging for tabular data annotation in general cases. The long term direction is to build a general tabular data annotations (MTab+ Figure 6.1). In particular, the problems of structure annotations and semantic annotations could be addressed in a unified system.

One of the other future directions is on document understanding where natural language text and data in document tables are both taken considered to improve document understanding. For example, the task of information retrieval on scientific publications is limited with keywords searching, and it is less useful for table searching (finding relevant tables contains a specific value). Or another application such as leader board summarization where studies methods and evaluation results from academic papers are automated processed and integrated into online leaderboard [1].

[1] https://paperswithcode.com/task/question-answering

Bibliography

[1] Kyosuke Nishida, Kugatsu Sadamitsu, Ryuichiro Higashinaka, and Yoshihiro Matsuo. Understanding the semantic structures of tables with a hybrid deep neural network architecture. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pages 168–174, 2017. URL http://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14396. xv, 14, 17, 18, 104

[2] S. K. Ramnandan, Amol Mittal, Craig A. Knoblock, and Pedro A. Szekely. Assigning semantic labels to data sources. In *The Semantic Web. Latest Advances and New Domains - 12th European Semantic Web Conference, ESWC 2015, Portoroz, Slovenia, May 31 - June 4, 2015. Proceedings*, pages 403–417, 2015. doi: 10.1007/978-3-319-18818-8_25. URL https://doi.org/10.1007/978-3-319-18818-8_25. xv, 23, 29, 30, 31, 37, 38, 46, 48, 52, 53, 54, 59, 61, 64, 67, 69, 70, 74, 109

[3] Yannis Charalabidis, Anneke Zuiderwijk, Charalampos Alexopoulos, Marijn Janssen, Thomas Lampoltshammer, and Enrico Ferro. *Open Data Directives and Policies*. Springer International Publishing, Cham, 2018. ISBN 978-3-319-90850-2. doi: 10.1007/978-3-319-90850-2_3. URL https://doi.org/10.1007/978-3-319-90850-2_3. 4

[4] Michael Färber, Frederic Bartscherer, Carsten Menne, and Achim Rettinger. Linked data quality of dbpedia, freebase, opencyc, wikidata, and YAGO. *Semantic Web*, 9 (1):77–129, 2018. doi: 10.3233/SW-170275. URL https://doi.org/10.3233/SW-170275. 4

[5] Sebastian Neumaier, Jürgen Umbrich, Josiane Xavier Parreira, and Axel Polleres.

Multi-level semantic labelling of numerical values. In *The Semantic Web - ISWC 2016 - 15th International Semantic Web Conference, Kobe, Japan, October 17-21, 2016, Proceedings, Part I*, pages 428–445, 2016. doi: 10.1007/978-3-319-46523-4_26. URL https://doi.org/10.1007/978-3-319-46523-4_26. 5, 23, 25, 29, 32, 46, 110

[6] Phuc Nguyen and Hideaki Takeda. Semantic labeling for numerical values: Distribution-based similarities. In *Proceedings of the 47 Special Interest Group for Semantic Web and Ontology, Ishigaki, Japan*, number 12, March 2019. URL http://id.nii.ac.jp/1004/00009725/. 7, 13, 25, 30, 109

[7] Phuc Nguyen, Khai Nguyen, Ryutaro Ichise, and Hideaki Takeda. Embnum+: Effective, efficient, and robust semantic labeling for numerical values. *New Generation Computing*, Nov 2019. ISSN 1882-7055. doi: 10.1007/s00354-019-00076-w. URL https://doi.org/10.1007/s00354-019-00076-w. 7, 13, 25, 70, 74, 88, 103, 109

[8] Phuc Nguyen, Natthawut Kertkeidkachorn, Ryutaro Ichise, and Hideaki Takeda. Mtab: Matching tabular data to knowledge graph using probability models. *CoRR*, abs/1910.00246, 2019. URL http://arxiv.org/abs/1910.00246. 7, 13, 25, 109, 110

[9] 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*, pages 545–554, 2011. doi: 10.1145/1935826.1935904. URL https://doi.org/10.1145/1935826.1935904. 18

[10] Michael J. Cafarella, Alon Y. Halevy, Yang Zhang, Daisy Zhe Wang, and Eugene Wu. Uncovering the relational web. In *11th International Workshop on the Web and Databases, WebDB 2008, Vancouver, BC, Canada, June 13, 2008*, 2008. URL http://webdb2008.como.polimi.it/images/stories/WebDB2008/paper30.pdf. 18, 19, 20

[11] Julian Eberius, Katrin Braunschweig, Markus Hentsch, Maik Thiele, Ahmad Ahmadov, and Wolfgang Lehner. Building the dresden web table corpus: A classification approach. In *2nd IEEE/ACM International Symposium on Big Data Computing, BDC 2015, Limassol, Cyprus, December 7-10, 2015*, pages 41–50, 2015. doi: 10.1109/BDC.2015.30. URL https://doi.org/10.1109/BDC.2015.30. 18

[12] David Pinto, Michael Branstein, Ryan G. Coleman, W. Bruce Croft, Matthew King, Wei Li, and Xing Wei. Quasm: a system for question answering using semi-structured data. In *ACM/IEEE Joint Conference on Digital Libraries, JCDL 2002, Portland, Oregon, USA, June 14-18, 2002, Proceedings*, pages 46–55, 2002. doi: 10.1145/544220.544228. URL https://doi.org/10.1145/544220.544228. 19, 20

[13] Dominique Ritze, Oliver Lehmberg, and Christian Bizer. Matching HTML tables to dbpedia. In *Proceedings of the 5th International Conference on Web Intelligence, Mining and Semantics, WIMS 2015, Larnaca, Cyprus, July 13-15, 2015*, pages 10:1–10:6, 2015. doi: 10.1145/2797115.2797118. URL https://doi.org/10.1145/2797115.2797118. 19, 20, 21, 22, 30, 46, 85

[14] Sung-Won Jung and Hyuk-Chul Kwon. A scalable hybrid approach for extracting head components from web tables. *IEEE Trans. Knowl. Data Eng.*, 18(2):174–187, 2006. doi: 10.1109/TKDE.2006.19. URL https://doi.org/10.1109/TKDE.2006.19. 19

[15] Rakesh Pimplikar and Sunita Sarawagi. Answering table queries on the web using column keywords. *PVLDB*, 5(10):908–919, 2012. doi: 10.14778/2336664.2336665. URL http://vldb.org/pvldb/vol5/p908_rakeshpimplikar_vldb2012.pdf. 19

[16] Girija Limaye, Sunita Sarawagi, and Soumen Chakrabarti. Annotating and searching web tables using entities, types and relationships. *PVLDB*, 3(1):1338–1347, 2010. doi: 10.14778/1920841.1921005. URL http://www.vldb.org/pvldb/vldb2010/pvldb_vol3/R118.pdf. 19, 21, 22, 23, 85

[17] Jingjing Wang, Haixun Wang, Zhongyuan Wang, and Kenny Qili Zhu. Understanding tables on the web. In *Conceptual Modeling - 31st International Conference ER 2012, Florence, Italy, October 15-18, 2012. Proceedings*, pages 141–155, 2012. doi: 10.1007/978-3-642-34002-4_11. URL https://doi.org/10.1007/978-3-642-34002-4_11. 19, 20, 22, 24, 31, 46

[18] Minoru Yoshida, Kentaro Torisawa, and Jun' ichi Tsujii. A method to integrate tables of the world wide web. In *Proceedings of the International Workshop on Web Document Analysis (WDA 2001)*, pages 31–34, 2001. 19

[19] Varish Mulwad, Tim Finin, and Anupam Joshi. Semantic message passing for generating linked data from tables. In *The Semantic Web - ISWC 2013 - 12th*

International Semantic Web Conference, Sydney, NSW, Australia, October 21-25, 2013, Proceedings, Part I, pages 363–378, 2013. doi: 10.1007/978-3-642-41335-3_23. URL https://doi.org/10.1007/978-3-642-41335-3_23. 19, 21, 22, 23

[20] Yalin Wang and Jianying Hu. Detecting tables in HTML documents. In *Document Analysis Systems V, 5th International Workshop, DAS 2002, Princeton, NJ, USA, August 19-21, 2002, Proceedings*, pages 249–260, 2002. doi: 10.1007/3-540-45869-7_29. URL https://doi.org/10.1007/3-540-45869-7_29. 20

[21] Ziqi Zhang. Effective and efficient semantic table interpretation using tableminer$^+$. *Semantic Web*, 8(6):921–957, 2017. doi: 10.3233/SW-160242. URL https://doi.org/10.3233/SW-160242. 20, 21, 22, 23, 24, 104

[22] Petros Venetis, Alon Y. Halevy, Jayant Madhavan, Marius Pasca, Warren Shen, Fei Wu, Gengxin Miao, and Chung Wu. Recovering semantics of tables on the web. *PVLDB*, 4(9):528–538, 2011. doi: 10.14778/2002938.2002939. URL http://www.vldb.org/pvldb/vol4/p528-venetis.pdf. 20, 31, 46

[23] Xiao Ling, Alon Y. Halevy, Fei Wu, and Cong Yu. Synthesizing union tables from the web. In *IJCAI 2013, Proceedings of the 23rd International Joint Conference on Artificial Intelligence, Beijing, China, August 3-9, 2013*, pages 2677–2683, 2013. URL http://www.aaai.org/ocs/index.php/IJCAI/IJCAI13/paper/view/6758. 21

[24] Oliver Lehmberg and Christian Bizer. Stitching web tables for improving matching quality. *PVLDB*, 10(11):1502–1513, 2017. doi: 10.14778/3137628.3137657. URL http://www.vldb.org/pvldb/vol10/p1502-lehmberg.pdf. 21, 104

[25] Yoones A. Sekhavat, Francesco Di Paolo, Denilson Barbosa, and Paolo Merialdo. Knowledge base augmentation using tabular data. In *Proceedings of the Workshop on Linked Data on the Web co-located with the 23rd International World Wide Web Conference (WWW 2014), Seoul, Korea, April 8, 2014*, 2014. URL http://ceur-ws.org/Vol-1184/ldow2014_paper_02.pdf. 21, 24, 29, 45

[26] Chandra Sekhar Bhagavatula, Thanapon Noraset, and Doug Downey. Tabel: Entity linking in web tables. In *The Semantic Web - ISWC 2015 - 14th International Semantic Web Conference, Bethlehem, PA, USA, October 11-15, 2015, Proceedings,*

Part I, pages 425–441, 2015. doi: 10.1007/978-3-319-25007-6_25. URL https://doi.org/10.1007/978-3-319-25007-6_25. 21, 24

[27] Sreeram Balakrishnan, Alon Y. Halevy, Boulos Harb, Hongrae Lee, Jayant Madhavan, Afshin Rostamizadeh, Warren Shen, Kenneth Wilder, Fei Wu, and Cong Yu. Applying web tables in practice. In *CIDR 2015, Seventh Biennial Conference on Innovative Data Systems Research, Asilomar, CA, USA, January 4-7, 2015, Online Proceedings*, 2015. 21, 22, 24

[28] Xu Chu, John Morcos, Ihab F. Ilyas, Mourad Ouzzani, Paolo Papotti, Nan Tang, and Yin Ye. KATARA: A data cleaning system powered by knowledge bases and crowdsourcing. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data, Melbourne, Victoria, Australia, May 31 - June 4, 2015*, pages 1247–1261, 2015. doi: 10.1145/2723372.2749431. URL https://doi.org/10.1145/2723372.2749431. 21, 22, 23

[29] Vasilis Efthymiou, Oktie Hassanzadeh, Mariano Rodriguez-Muro, and Vassilis Christophides. Matching web tables with knowledge base entities: From entity lookups to entity embeddings. In *The Semantic Web - ISWC 2017 - 16th International Semantic Web Conference, Vienna, Austria, October 21-25, 2017, Proceedings, Part I*, pages 260–277, 2017. doi: 10.1007/978-3-319-68288-4_16. URL https://doi.org/10.1007/978-3-319-68288-4_16. 21, 104

[30] Patrice Buche, Juliette Dibie-Barthélemy, Liliana Ibanescu, and Lydie Soler. Fuzzy web data tables integration guided by an ontological and terminological resource. *IEEE Trans. Knowl. Data Eng.*, 25(4):805–819, 2013. doi: 10.1109/TKDE.2011.245. URL https://doi.org/10.1109/TKDE.2011.245. 22, 23

[31] Ju Fan, Meiyu Lu, Beng Chin Ooi, Wang-Chiew Tan, and Meihui Zhang. A hybrid machine-crowdsourcing system for matching web tables. In *IEEE 30th International Conference on Data Engineering, Chicago, ICDE 2014, IL, USA, March 31 - April 4, 2014*, pages 976–987, 2014. doi: 10.1109/ICDE.2014.6816716. URL https://doi.org/10.1109/ICDE.2014.6816716. 22

[32] Jiaoyan Chen, Ernesto Jiménez-Ruiz, Ian Horrocks, and Charles A. Sutton. Learning semantic annotations for tabular data. In *Proceedings of the Twenty-*

Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019, pages 2088–2094, 2019. doi: 10.24963/ijcai.2019/289. URL https://doi.org/10.24963/ijcai.2019/289. 22

[33] Jiaoyan Chen, Ernesto Jiménez-Ruiz, Ian Horrocks, and Charles A. Sutton. Colnet: Embedding the semantics of web tables for column type prediction. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence, January 27 - February 1, 2019, Honolulu, Hawaii, USA*, pages 29–36, 2019. doi: 10.1609/aaai.v33i01.330129. URL https://doi.org/10.1609/aaai.v33i01.330129. 22

[34] Emir Muñoz, Aidan Hogan, and Alessandra Mileo. Triplifying wikipedia's tables. In *Proceedings of the First International Workshop on Linked Data for Information Extraction (LD4IE 2013) co-located with the 12th International Semantic Web Conference (ISWC 2013), Sydney, Australia, October 21, 2013*, 2013. URL http://ceur-ws.org/Vol-1057/MunozEtAl_LD4IE2013.pdf. 23

[35] Michael Stonebraker, Daniel Bruckner, Ihab F. Ilyas, George Beskales, Mitch Cherniack, Stanley B. Zdonik, Alexander Pagan, and Shan Xu. Data curation at scale: The data tamer system. In *CIDR 2013, Sixth Biennial Conference on Innovative Data Systems Research, Asilomar, CA, USA, January 6-9, 2013, Online Proceedings*, 2013. URL http://cidrdb.org/cidr2013/Papers/CIDR13_Paper28.pdf. 23, 31, 46

[36] Minh Pham, Suresh Alse, Craig A. Knoblock, and Pedro A. Szekely. Semantic labeling: A domain-independent approach. In *The Semantic Web - ISWC 2016 - 15th International Semantic Web Conference, Kobe, Japan, October 17-21, 2016, Proceedings, Part I*, pages 446–462, 2016. doi: 10.1007/978-3-319-46523-4_27. URL https://doi.org/10.1007/978-3-319-46523-4_27. 23, 29, 30, 31, 37, 38, 46, 48, 59, 64, 66, 67, 69, 70, 74, 109

[37] Xiaoxin Yin, Wenzhao Tan, and Chao Liu. FACTO: a fact lookup engine based on web tables. In *Proceedings of the 20th International Conference on World Wide Web, WWW 2011, Hyderabad, India, March 28 - April 1, 2011*, pages 507–516, 2011. doi: 10.1145/1963405.1963477. URL https://doi.org/10.1145/1963405.1963477. 23, 24

[38] Fernando Chirigati, Jialu Liu, Flip Korn, You Wu, Cong Yu, and Hao Zhang. Knowledge exploration using tables on the web. *PVLDB*, 10(3):193–204, 2016. doi: 10.14778/3021924.3021935. URL http://www.vldb.org/pvldb/vol10/p193-chirigati.pdf. 24

[39] Mohamed Yakout, Kris Ganjam, Kaushik Chakrabarti, and Surajit Chaudhuri. Infogather: entity augmentation and attribute discovery by holistic matching with web tables. In *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2012, Scottsdale, AZ, USA, May 20-24, 2012*, pages 97–108, 2012. doi: 10.1145/2213836.2213848. URL https://doi.org/10.1145/2213836.2213848. 24

[40] Oliver Lehmberg, Dominique Ritze, Petar Ristoski, Robert Meusel, Heiko Paulheim, and Christian Bizer. The mannheim search join engine. *J. Web Semant.*, 35: 159–166, 2015. doi: 10.1016/j.websem.2015.05.001. URL https://doi.org/10.1016/j.websem.2015.05.001. 24

[41] Matteo Cannaviccio, Lorenzo Ariemma, Denilson Barbosa, and Paolo Merialdo. Leveraging wikipedia table schemas for knowledge graph augmentation. In *Proceedings of the 21st International Workshop on the Web and Databases, Houston, TX, USA, June 10, 2018*, pages 5:1–5:6, 2018. doi: 10.1145/3201463.3201468. URL https://doi.org/10.1145/3201463.3201468. 24

[42] Dominique Ritze. *Web-Scale Web Table to Knowledge Base Matching*. PhD thesis, University of Mannheim, Germany, 2017. URL https://ub-madoc.bib.uni-mannheim.de/43123. 24, 104

[43] Oliver Lehmberg. *Web table integration and profiling for knowledge base augmentation*. PhD thesis, University of Mannheim, Germany, Mannheim, 2019. URL https://madoc.bib.uni-mannheim.de/52346/. 24

[44] Besnik Fetahu, Avishek Anand, and Maria Koutraki. Tablenet: An approach for determining fine-grained relations for wikipedia tables. In *The World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019*, pages 2736–2742, 2019. doi: 10.1145/3308558.3313629. URL https://doi.org/10.1145/3308558.3313629. 24

[45] Huan Sun, Hao Ma, Xiaodong He, Wen-tau Yih, Yu Su, and Xifeng Yan. Table cell search for question answering. In *Proceedings of the 25th International Conference on World Wide Web, WWW 2016, Montreal, Canada, April 11 - 15, 2016*, pages 771–782, 2016. doi: 10.1145/2872427.2883080. URL https://doi.org/10.1145/2872427.2883080. 24

[46] Sunita Sarawagi and Soumen Chakrabarti. Open-domain quantity queries on web tables: annotation, response, and consensus models. In *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA - August 24 - 27, 2014*, pages 711–720, 2014. doi: 10.1145/2623330.2623749. URL https://doi.org/10.1145/2623330.2623749. 24

[47] Thanh Tam Nguyen, Nguyen Quoc Viet Hung, Matthias Weidlich, and Karl Aberer. Result selection and summarization for web table search. In *31st IEEE International Conference on Data Engineering, ICDE 2015, Seoul, South Korea, April 13-17, 2015*, pages 231–242, 2015. doi: 10.1109/ICDE.2015.7113287. URL https://doi.org/10.1109/ICDE.2015.7113287. 29, 45

[48] Fatemeh Nargesian, Erkang Zhu, Ken Q. Pu, and Renée J. Miller. Table union search on open data. *PVLDB*, 11(7):813–825, 2018. doi: 10.14778/3192965.3192973. URL http://www.vldb.org/pvldb/vol11/p813-nargesian.pdf. 29, 45

[49] Oliver Lehmberg, Dominique Ritze, Petar Ristoski, Robert Meusel, Heiko Paulheim, and Christian Bizer. The mannheim search join engine. *J. Web Semant.*, 35: 159–166, 2015. doi: 10.1016/j.websem.2015.05.001. URL https://doi.org/10.1016/j.websem.2015.05.001. 29, 45

[50] Ahmad Ahmadov, Maik Thiele, Julian Eberius, Wolfgang Lehner, and Robert Wrembel. Towards a hybrid imputation approach using web tables. In *2nd IEEE/ACM International Symposium on Big Data Computing, BDC 2015, Limassol, Cyprus, December 7-10, 2015*, pages 21–30, 2015. doi: 10.1109/BDC.2015.38. URL https://doi.org/10.1109/BDC.2015.38. 29, 45

[51] Meihui Zhang and Kaushik Chakrabarti. Infogather+: semantic matching and annotation of numeric and time-varying attributes in web tables. In *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD*

2013, New York, NY, USA, June 22-27, 2013, pages 145–156, 2013. doi: 10.1145/2463676.2465276. URL https://doi.org/10.1145/2463676.2465276. 29, 30, 45, 46

[52] Xin Dong, Evgeniy Gabrilovich, Geremy Heitz, Wilko Horn, Ni Lao, Kevin Murphy, Thomas Strohmann, Shaohua Sun, and Wei Zhang. Knowledge vault: a web-scale approach to probabilistic knowledge fusion. In *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA - August 24 - 27, 2014*, pages 601–610, 2014. doi: 10.1145/2623330.2623623. URL https://doi.org/10.1145/2623330.2623623. 29, 45

[53] Ronald L. Wasserstein and Nicole A. Lazar. The asa's statement on p -values: Context, process, and purpose. *The American Statistician*, 70(2):129–133, 2016. URL https://EconPapers.repec.org/RePEc:taf:amstat:v:70:y:2016:i:2:p:129-133. 30

[54] Sander Greenland, Stephen J Senn, Kenneth J Rothman, John B Carlin, Charles Poole, Steven N Goodman, and Douglas G Altman. Statistical tests, p values, confidence intervals, and power: a guide to misinterpretations. *European journal of epidemiology*, 31(4):337–350, 2016. 30

[55] Monya Baker. Statisticians issue warning over misuse of p values. *Nature News*, 531(7593):151, 2016. 30

[56] Ivan Ermilov, Sören Auer, and Claus Stadler. User-driven semantic mapping of tabular data. In *I-SEMANTICS 2013 - 9th International Conference on Semantic Systems, ISEM '13, Graz, Austria, September 4-6, 2013*, pages 105–112, 2013. doi: 10.1145/2506182.2506196. URL https://doi.org/10.1145/2506182.2506196. 30, 46

[57] Marco D. Adelfio and Hanan Samet. Schema extraction for tabular data on the web. *PVLDB*, 6(6):421–432, 2013. doi: 10.14778/2536336.2536343. URL http://www.vldb.org/pvldb/vol6/p421-adelfio.pdf. 31, 46

[58] Erich Leo Lehmann and Joseph P. Romano. *Testing Statistical Hypotheses, Third Edition*. Springer texts in statistics. Springer, 2008. ISBN 978-0-387-98864-1. doi: 10.1007/0-387-27605-X. URL https://doi.org/10.1007/0-387-27605-X. 31

[59] Markus Neuhäuser. *Wilcoxon-Mann-Whitney Test*, pages 1656–1658. Springer, 2011. doi: 10.1007/978-3-642-04898-2_615. URL https://doi.org/10.1007/978-3-642-04898-2_615. 31

[60] Wikipedia contributors. Inverse transform sampling — Wikipedia, the free encyclopedia, 2018. [Online; accessed 3-July-2018]. 34, 48

[61] Oliver Lehmberg, Dominique Ritze, Robert Meusel, and Christian Bizer. A large public corpus of web tables containing time and context metadata. In *Proceedings of the 25th International Conference on World Wide Web, WWW 2016, Montreal, Canada, April 11-15, 2016, Companion Volume*, pages 75–76, 2016. doi: 10.1145/2872518.2889386. URL https://doi.org/10.1145/2872518.2889386. 45, 104

[62] Johann Mitlöhner, Sebastian Neumaier, Jürgen Umbrich, and Axel Polleres. Characteristics of open data CSV files. In *2nd International Conference on Open and Big Data, OBD 2016, Vienna, Austria, August 22-24, 2016*, pages 72–79, 2016. doi: 10.1109/OBD.2016.18. URL https://doi.org/10.1109/OBD.2016.18. 45, 46, 48

[63] Yoshua Bengio. Learning deep architectures for AI. *Foundations and Trends in Machine Learning*, 2(1):1–127, 2009. doi: 10.1561/2200000006. URL https://doi.org/10.1561/2200000006. 47

[64] Matthew D. Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I*, pages 818–833, 2014. doi: 10.1007/978-3-319-10590-1_53. URL https://doi.org/10.1007/978-3-319-10590-1_53. 47

[65] Pierre Sermanet, David Eigen, Xiang Zhang, Michaël Mathieu, Rob Fergus, and Yann LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014. URL http://arxiv.org/abs/1312.6229. 47

[66] Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. Deepface: Closing the gap to human-level performance in face verification. In *2014 IEEE*

Conference on Computer Vision and Pattern Recognition, CVPR 2014, Columbus, OH, USA, June 23-28, 2014, pages 1701–1708, 2014. doi: 10.1109/CVPR.2014.220. URL https://doi.org/10.1109/CVPR.2014.220. 47

[67] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 815–823, 2015. doi: 10.1109/CVPR.2015.7298682. URL https://doi.org/10.1109/CVPR.2015.7298682. 47, 55

[68] Weihua Chen, Xiaotang Chen, Jianguo Zhang, and Kaiqi Huang. Beyond triplet loss: A deep quadruplet network for person re-identification. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 1320–1329, 2017. doi: 10.1109/CVPR.2017.145. URL https://doi.org/10.1109/CVPR.2017.145. 47

[69] Phuc Nguyen, Khai Nguyen, Ryutaro Ichise, and Hideaki Takeda. Embnum: Semantic labeling for numerical values with deep metric learning. In *Semantic Technology - 8th Joint International Conference, JIST 2018, Awaji, Japan, November 26-28, 2018, Proceedings*, pages 119–135, 2018. doi: 10.1007/978-3-030-04284-4_9. URL https://doi.org/10.1007/978-3-030-04284-4_9. 52

[70] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL http://arxiv.org/abs/1409.1556. 55

[71] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 770–778, 2016. doi: 10.1109/CVPR.2016.90. URL https://doi.org/10.1109/CVPR.2016.90. 55, 56

[72] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, pages 448–456, 2015. URL http://proceedings.mlr.press/v37/ioffe15.html. 66

[73] Robyn Speer. ftfy. Zenodo, 2019. URL https://github.com/LuminosoInsight/python-ftfy. Version 5.5. 86

[74] Edouard Grave, Tomas Mikolov, Armand Joulin, and Piotr Bojanowski. Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2017, Valencia, Spain, April 3-7, 2017, Volume 2: Short Papers*, pages 427–431, 2017. URL https://www.aclweb.org/anthology/E17-2068/. 86

[75] Matthew Honnibal and Ines Montani. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear, 2017. URL https://spacy.io/. 87

[76] Oktie Hassanzadeh, Vasilis Efthymiou, Jiaoyan Chen, Ernesto Jiménez-Ruiz, and Kavitha Srinivas. SemTab2019: Semantic Web Challenge on Tabular Data to Knowledge Graph Matching - 2019 Data Sets, October 2019. URL https://doi.org/10.5281/zenodo.3518539. 94, 97

[77] Gilles Vandewiele, Bram Steenwinckel, Filip De Turck, and Femke Ongenae. Cvs2kg: Transforming tabular data into semantic knowledge. Semantic Web Challenge on Tabular Data to Knowledge Graph Matching, ISWC, 2019. URL http://www.cs.ox.ac.uk/isg/challenges/sem-tab/papers/IDLab.pdf. 102

[78] Avijit Thawani, Minda Hu, Erdong Hu, Husain Zafar, Naren Teja Divvala, Amandeep Singh, Ehsan Qasemi, Pedro Szekely, and Jay Pujara. Entity linking to knowledge graphs to infer column types and properties. Semantic Web Challenge on Tabular Data to Knowledge Graph Matching, ISWC, 2019. URL http://www.cs.ox.ac.uk/isg/challenges/sem-tab/papers/Tabularisi.pdf. 102

[79] Marco Cremaschi, Roberto Avogadro, and David Chieregato. Mantistable: an automatic approach for the semantic table interpretation. Semantic Web Challenge on Tabular Data to Knowledge Graph Matching, ISWC, 2019. URL http://www.cs.ox.ac.uk/isg/challenges/sem-tab/papers/MantisTable.pdf. 102

[80] Yoan Chabot, Thomas Labbe, Jixiong Liu, and Raphaël Troncy. Dagobah: An end-to-end context-free tabular data semantic annotation system. Semantic Web

Challenge on Tabular Data to Knowledge Graph Matching, ISWC, 2019. URL http://www.cs.ox.ac.uk/isg/challenges/sem-tab/papers/DAGOBAH.pdf. 103

[81] Hiroaki Morikawa, Fumihito Nishino, and Nobuyuki Igata. Semantic table interpretation using lod4all. Semantic Web Challenge on Tabular Data to Knowledge Graph Matching, ISWC, 2019. URL http://www.cs.ox.ac.uk/isg/challenges/sem-tab/papers/LOD4ALL.pdf. 103

[82] Daniela Oliveira and Mathieu d'Aquin. Adog - anotating data with ontologies and graphs. Semantic Web Challenge on Tabular Data to Knowledge Graph Matching, ISWC, 2019. URL http://www.cs.ox.ac.uk/isg/challenges/sem-tab/papers/ADOG.pdf. 103

[83] Maximilian Nickel and Douwe Kiela. Poincaré embeddings for learning hierarchical representations. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 6338–6347, 2017. URL http://papers.nips.cc/paper/7213-poincare-embeddings-for-learning-hierarchical-representations. 110