

MASTER

**Beyond Equivalence**

**The Role of Domain Knowledge in Ontology Mapping Refinement**

Snijder, Lucas L.

*Award date:*  
2024

[Link to publication](#)

**Disclaimer**

This document contains a student thesis (bachelor's or master's), as authored by a student at Eindhoven University of Technology. Student theses are made available in the TU/e repository upon obtaining the required degree. The grade received is not published on the document as presented in the repository. The required complexity or quality of research of student theses may vary by program, and the required minimum study period may vary in duration.

**General rights**

Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain

**Take down policy**

If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

# Beyond Equivalence: The Role of Domain Knowledge in Ontology Mapping Refinement

*Master's thesis*

*Author*

Lucas L. Snijder

*Graduation supervisor*

George H. L. Fletcher

*Graduation co-supervisor*

Romy van Drie

*Graduation co-supervisor*

Maike de Boer

*Assessment committee member*

Natalia Sidorova

February 7, 2024

# Contents

<b>1</b>	<b>Introduction</b>	<b>3</b>
1.1	Motivation . . . . .	3
1.2	Current solutions . . . . .	5
1.3	Challenges . . . . .	6
1.4	Research questions . . . . .	6
1.5	Application of this research . . . . .	7
1.6	Outline and main contributions of the thesis . . . . .	7
1.7	Terminology . . . . .	8
<b>2</b>	<b>Literature Analysis of Ontology Alignment with Extended Relations</b>	<b>9</b>
2.1	Ontology matching . . . . .	9
2.1.1	Classifying ontology matching methods . . . . .	9
2.1.2	Language-based techniques . . . . .	10
2.1.3	Non-contextual word embeddings . . . . .	10
2.1.4	Contextual word embeddings . . . . .	11
2.1.5	Generative large language models . . . . .	11
2.2	Ontology alignment with extended relations . . . . .	12
2.2.1	One-step approaches . . . . .	12
2.2.2	Two-step approaches (mapping refinement) . . . . .	13
2.3	Knowledge graph completion . . . . .	13
2.3.1	Relation between ontologies and knowledge graphs . . . . .	14
2.3.2	Relation between relation prediction and mapping refinement . . . . .	14
<b>3</b>	<b>Defining the Mapping Refinement Problem</b>	<b>16</b>
3.1	Ontologies . . . . .	16
3.1.1	Defining ontologies . . . . .	16
3.1.2	Ontology languages . . . . .	17
3.2	Ontology alignment . . . . .	17
3.2.1	Definition of ontology matching . . . . .	18
3.2.2	Definition of mapping refinement . . . . .	18
3.2.3	Semantic relationship types of a refined mapping . . . . .	18
<b>4</b>	<b>Methods</b>	<b>20</b>
4.1	STROMA . . . . .	20
4.1.1	Type computation strategies . . . . .	20
4.1.2	Verification step . . . . .	21
4.1.3	Selection step . . . . .	22
4.2	Domain knowledge methods . . . . .	22
4.2.1	BERT . . . . .	22
4.2.2	Mapping refinement methods that use BERT . . . . .	26
4.3	The GPT-4 method . . . . .	28
4.3.1	GPT-4 . . . . .	28

4.3.2	Mapping refinement with GPT-4 . . . . .	29
<b>5</b>	<b>Empirical Evaluation</b>	<b>31</b>
5.1	Experimental methodology . . . . .	31
5.1.1	Experiments . . . . .	31
5.1.2	Use cases . . . . .	32
5.1.3	Data preparation . . . . .	36
5.1.4	Evaluation Methods . . . . .	37
5.2	Experimental results . . . . .	38
5.2.1	Experiment 1: Comparing the current state-of-the-art methods that do not use domain knowledge to those that do . . . . .	38
5.2.2	Experiment 2: Validating TaSeR . . . . .	39
5.2.3	Experiment 3: Comparing methods that use domain knowledge and GPT-4 method in a zero-shot setting . . . . .	41
5.2.4	Experiment 4: Comparing GPT-4 in a zero-shot and a few-shot setting	43
<b>6</b>	<b>Discussion</b>	<b>45</b>
6.1	Discussion of the experimental results . . . . .	45
6.1.1	SQ1: How do current mapping refinement methods that incorporate domain knowledge compare to methods that do not? . . . . .	45
6.1.2	SQ2: Does including additional general knowledge via pre-fine-tuning, as done in the current state of the art of mapping refinement, benefit its performance? . . . . .	45
6.1.3	SQ3: How do state-of-the-art mapping refinement methods compare to generative large language models in a zero-shot setting? . . . . .	46
6.1.4	SQ4: Does incorporating domain knowledge into generative large language models through few-shot learning enhance their performance in mapping refinement? . . . . .	47
6.2	Discussion of the application of mapping refinement . . . . .	47
<b>7</b>	<b>Conclusion</b>	<b>49</b>
7.1	Summary of the results . . . . .	49
7.2	Summary of the main contributions . . . . .	49
7.2.1	Implications for Researchers . . . . .	49
7.2.2	Implications for Practitioners . . . . .	50
7.3	Limitations . . . . .	50
7.4	Future work . . . . .	51
	<b>Bibliography</b>	<b>52</b>
	<b>A Full experiments table</b>	<b>64</b>

# Abstract

In this thesis we explore various approaches to mapping refinement in the web, labor market and biomedical domains. The motivation comes from the labor market domain, where skill and occupation ontologies have to be interoperable. This interoperability requires creating links between the ontologies, called ontology alignment. Part of creating a rich alignment is determining the semantic relationship of the links, called mapping refinement. However, accurately identifying these relationships remains a significant challenge. Current state-of-the-art methods exist of training a model using domain knowledge. With the recent rise of generative large language models like GPT-4, many state-of-the-art solutions for natural language processing tasks have been outperformed by these new models. This raises the question what the potential of these generative large language models is for mapping refinement and if the training using domain knowledge is still required for efficient mapping refinement. In this thesis we have validated and compared the current state-of-the-art methods STROMA and TaSeR and proposed a GPT-4 based approach. The main research question we tried to answer was whether domain knowledge is still required for effective mapping refinement in the age of generative large language models. The results indicated that the integration of domain knowledge in language models continues to outperform solely relying on generative large language models. This study provides insights into the current limitations and strengths of both pre-trained language models and generative large language models, underscoring the critical role of domain knowledge in mapping refinement processes. Both the code and the data can be found here: <https://github.com/lucasnijder/MapRef>.

# Acknowledgements

I would like to extend my sincere gratitude to everyone who made it possible for me to complete this thesis. Special thanks are due to Professor George Fletcher, who generously agreed to supervise my project mid-way. His guidance has been invaluable. My appreciation also goes to Romy van Drie, for her efforts in orchestrating a new project for me after the initial one did not materialize, and to Maaïke de Boer, for facilitating a swift transition to this new project. Their advice and guidance have been fundamental to my progress. I am grateful to Nina Donker for her support and for helping me maintain my sanity during the intense periods of thesis writing. Equally, I owe a debt of gratitude to my parents, whose unwavering support and provision of a peaceful place to work were crucial in the final weeks. I would also like to acknowledge the financial support provided by the Vaardig met Vaardigheden project, which was essential for my research. Lastly, a note of thanks to Maureen Snijder and Gijs Rijckaerd for their assistance in refining the text of my thesis.

# Chapter 1

## Introduction

### 1.1 Motivation

One of the challenges in our current (European) society is the discrepancy between supply and demand in the labor market [34]. While there are many vacancies that are open and there are many unemployed people, they seem to be unable to find a match. One of the approaches to help the demand and supply problem in the labor market is to change from degree based hiring towards skills based hiring [16]. These skills based approaches ask for a common and up-to-date skills language that describes how occupations and skills relate to achieve their full potential. By developing a domain-specific language for the labor market, job discovery can be simplified, allowing individuals to be assessed based on their skills potential rather than being constrained by their current or past job roles. Labor markets vary significantly among countries, necessitating the development of specific labor ontologies instead of a one-size-fits-all solution. Languages like these have been developed in recent years in the form of ontologies, like ESCO<sup>1</sup> for the European region, O\*NET<sup>2</sup> for North America and recently the development started on CompetentNL<sup>3</sup> for the Netherlands. As an example of these ontologies, a subset of the ESCO and O\*NET ontologies can be seen in Figure 1.1. However, as each region has its own ontology, problems arise. Varied ontologies hinder interoperability, making data and information sharing across international borders difficult. Exactly this information sharing is of paramount interest as the Institute for Employee Insurance (UWV<sup>4</sup>) unveiled a study that stated that despite certain job categories grappling with shortages in the Netherlands, a whopping 70% of these roles have an oversupply in other EU nations [98]. This shows the potential of employing international workers. However, often misaligned expectations of both potential employees as well as employers lead to international hiring being tough.

The solution to these problems is to make the national and regional ontologies interoperable. The technical term for aligning such ontologies is called ontology alignment. Ontology alignment entails creating links between two ontologies, to map how entities from one relate to entities in the other. These links are called correspondences and the collection of these correspondences is called a mapping. The links between the two ontologies in Figure 1.1 represent such a mapping. Currently, most alignments are created by hand, which is tedious work as for each entity in one ontology, all entities in the to-be-aligned ontology need to be evaluated. For example, for the occupation *ICT system analyst* in ESCO, all occupations in O\*NET need to be checked whether they are similar. Besides finding matches, it is necessary to identify the semantic relationship between the entities in each established correspondence. This makes sure subtle difference between concepts can be modelled accurately. For example

---

<sup>1</sup><https://esco.ec.europa.eu>

<sup>2</sup><https://www.onetonline.org>

<sup>3</sup><https://www.competentnl.nl>

<sup>4</sup><https://www.uwv.nl/particulieren/>

*computer vision engineer* in ESCO and *mathematicians* in O\*NET have a subclass relation in the form of an *is-a* relation since *computer vision engineer* is a *mathematician* but not the other way around. A "subclass relation" refers to the hierarchical relationship between entities, where one entity (the superclass) encompasses or includes another entity (the subclass).

As an example of the full process of creating a refined mapping, we will look at Figure 1.1. The O\*NET Ontology has the occupations *Computer system analyst*, *Database administrators*, *Computer networks architects*, *Data scientists*, *Mathematicians* and *Operations research analysts*. The ESCO ontology has the occupations *ICT system analyst*, *ICT system architect*, *ICT consultant*, *Data scientist* and *Computer vision engineer*. In the first step, matching, we find the correspondences (*ICT system analyst*, *Computer system analysts*), (*ICT system architect*, *Computer network architects*), (*Data scientist*, *Data scientists*), (*Data scientist*, *Mathematicians*), (*Computer vision engineer*, *Mathematicians*). Note that *ICT consultants* from the ESCO ontology is not included since there is no similar entity in the O\*NET ontology. In the second step, mapping, we find the mappings (*ICT system analyst*, *Equivalent to*, *Computer system analysts*), (*ICT system architect*, *Inverse is-a*, *Computer network architects*), (*Data scientist*, *Equivalent to*, *Data scientists*), (*Data scientist*, *Is-a*, *Mathematicians*), (*Computer vision engineer*, *Is-a*, *Mathematicians*). This example is an easy one, as there are only a small number of occupations per ontology. But as ontologies are often thousands of entities large, creating a refined mapping is tedious work.

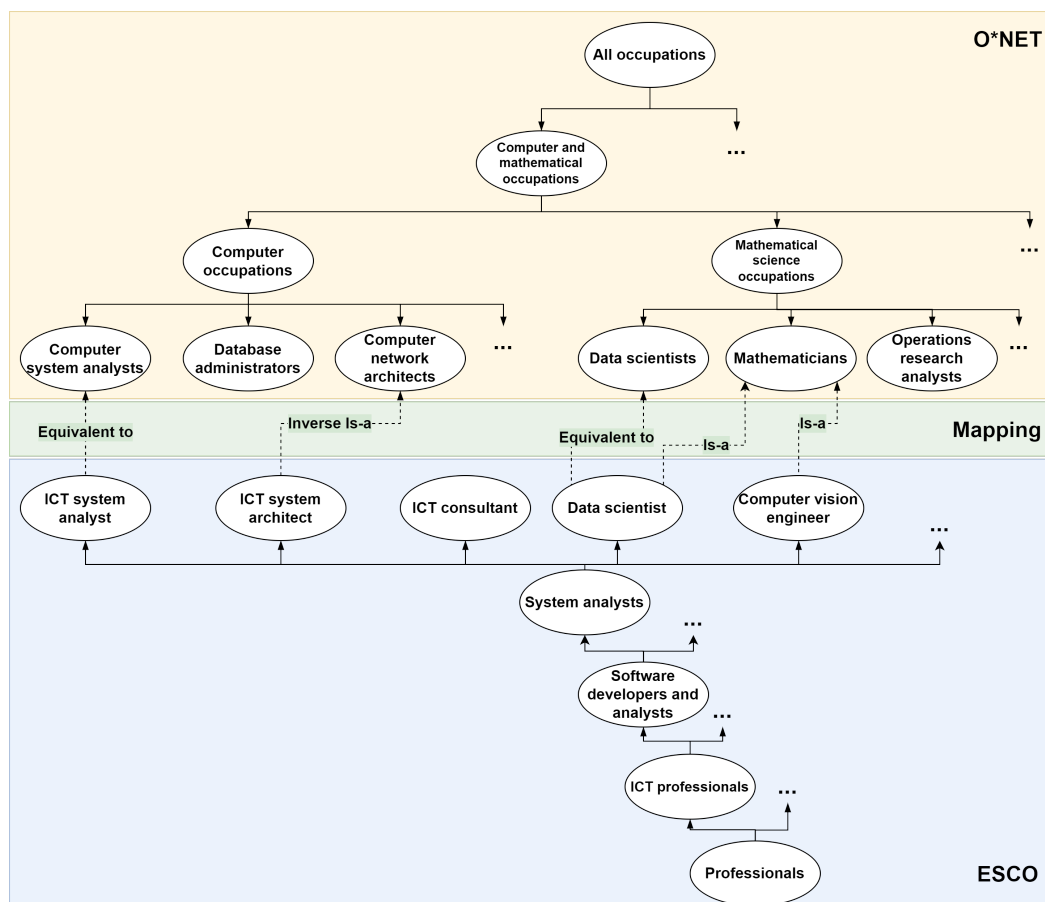


Figure 1.1: Example of a mapping between subsets of the ESCO and O\*NET ontologies

## 1.2 Current solutions

Since the 1980's there has been a lot of research into creating methods that can automatically create a mapping [68, 85, 89, 57, 99]. However, these methods focused only on establishing a correspondence between the entities, not determining what the relation between the entities is. While under-researched, there have been publications on ontology matching with extended relations such as *is-a* or *part-of* [32, 53, 101, 117]. These approaches are called one-step approaches for creating a refined mapping [7], as they combine matching and determining the relations in one method. In Figure 1.2 the structure of a one-step approach is shown.

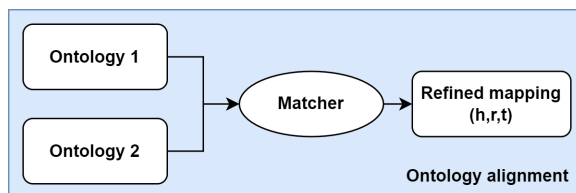


Figure 1.2: A simplified visualisation of one-step ontology alignment

A different approach to creating a refined mapping is not to try to find the specific relations during the matching process like the one-step approach, but to first find equivalence correspondences and afterwards refine these rough matches into refined correspondences with extended relations. These techniques are called mapping refinement techniques [7]. This combination of ontology matching and mapping refinement is the two-step approach to creating a refined mapping. Two-step approaches offer advantages over one-step approaches in terms of simplicity and generic applicability [7]. The simplicity of the two-step approach is found in its reduced search space compared to one-step approaches. One-step approaches need to evaluate all combinations of source and target entities, as well as all possible relations. This leads to a huge search space in which finding true correspondences is hard. The two-step approach not only improves match quality and efficiency by focusing on a more manageable set of correspondences, but also enhances flexibility by allowing the integration of additional relation types at later stages without redesigning the entire matching method. Furthermore, two-step methods facilitate compatibility with various ontology matching techniques, benefiting from the active development within the field. However, the effectiveness of two-step approaches heavily depends on the initial match's completeness and quality, emphasizing the need for robust methods in the matching phase. In Figure 1.3 the structure of a two-step approach is highlighted.

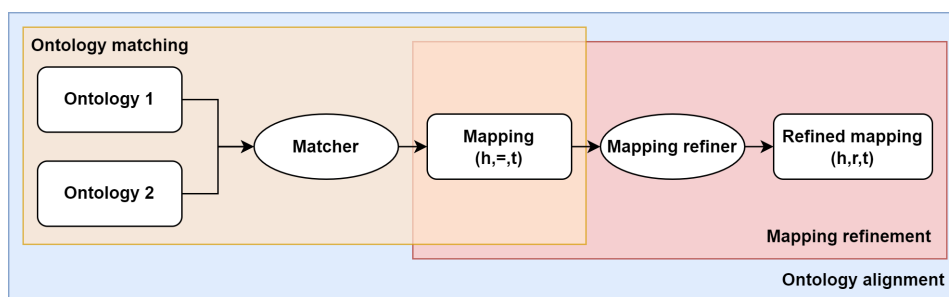


Figure 1.3: A simplified visualisation of two-step ontology alignment

While the two-step approach is a robust approach to creating a refined mapping, this area of research has not been studied extensively. The first mapping refinement method was STROMA [7], a rule-based system. Following this first publications, new methods have been proposed that make use of word embeddings. The first publication to propose this

was by Dhouib et al. [104]. Later Hertling and Paulheim proposed TaSeR [45], the current state-of-the-art method. It uses domain knowledge to fine-tune a BERT language model [24] which is then able to identify the relation between two given entities. In section 2 we will dive deeper into the existing literature.

### 1.3 Challenges

Despite significant advancements in ontology mapping and refinement methods, they still face challenges. Firstly, existing mapping refinement approaches, including the state-of-the-art TaSeR method, have not been extensively tested across diverse datasets. This limitation raises concerns about their generalizability in varying contexts. Additionally, our preliminary attempts to reproduce results from the TaSeR paper reveal that the performance is not equally good for all types of relations. This inconsistency in performance underscores the need for further investigation into the reliability of the TaSeR method. Moreover, the effectiveness of the TaSeR method heavily relies on the presence of subclass relations within the source and target ontologies. This dependency highlights a significant gap in the methodology, as it assumes there are enough subclass relations available for effectively learning the characteristics.

Interestingly, the broader field of natural language processing (NLP) has witnessed a paradigm shift with the rise of generative large language models such as GPT-4, which have started to outperform traditional domain knowledge-based approaches in tasks such as document classification [18], sentiment analysis [29] and text classification [103, 31]. This trend suggests that the necessity of domain knowledge might be diminishing, as generative large language models demonstrate remarkable capabilities in understanding and generating human-like text without explicit domain-specific tuning.

### 1.4 Research questions

In the previous section we have shown a quick overview of the current literature, unveiling that domain knowledge based methods are being replaced by generative large language models that do not require domain knowledge. Based on these recent advancements, we will formulate the research questions in this section. We will focus on the second step of the two-step system: mapping refinement. The main research question that we will try to answer in this thesis is:

*RQ: Is domain knowledge still required for mapping refinement in the age of generative large language models?*

We will try to answer the main research question by answering four subquestions. The goal of the first subquestion is to find out what the differences in performance are between current methods that do not use domain knowledge and those that do. The subquestion is formulated as:

*SQ1: How do current mapping refinement methods that incorporate domain knowledge compare to methods that do not?*

The second subquestion focuses on validating the current state-of-the-art method that uses domain knowledge, called TaSeR. As mentioned before, our preliminary experiments showed that the method did not work equally good for all covered relations. Additionally, it makes use of pre-fine-tuning as an extra step before fine-tuning for mapping refinement. In literature there are some examples of pre-fine-tuning, but these methods either continue pre-training on a task/domain specific dataset [102, 38], use very similar tasks as the pre-training tasks [114], or use multi-task training to enhance the language understanding of the model [4]. The goal of pre-fine-tuning in these papers is to enhance the language understanding of

the model. Pre-fine-tuning on a different but similar classification task as the one from the fine-tuning stage like in the TaSeR method has not yet been proposed in literature. Also, in the original TaSeR paper it was not tested whether adding the pre-fine-tuning step was beneficial for the performance [45]. The goal of the second subquestion is to validate the TaSeR method to find out whether the form of pre-fine-tuning as it is included is beneficial for the performance. Note that the dataset used for pre-fine-tuning is not regarded as domain knowledge as they are general knowledge sources. The subquestion is formulated as:

*SQ2: Does including additional general knowledge via pre-fine-tuning, as done in the current state of the art of mapping refinement, benefit its performance?*

The goal of the third subquestion is to find out whether the hassle of incorporating domain knowledge is still necessary to effectively perform mapping refinement, or that it can be done more efficiently by using the state-of-the-art generative large language model GPT-4. We focus on GPT-4 because many evaluation reports have come to similar conclusions that GPT-4 is the state-of-the-art generative large language model [84, 36]. The subquestion is formulated as:

*SQ3: How do state-of-the-art mapping refinement methods compare to generative large language models in a zero-shot setting?*

The goal of the fourth subquestion is to find out whether GPT-4 can benefit from domain knowledge, or that it might not be worth the hassle to incorporate it. The subquestion is formulated as:

*SQ4: Does incorporating domain knowledge into generative large language models through few-shot learning enhance their performance in mapping refinement?*

In this thesis, we assume the first step, ontology matching, has already been performed and an equivalence mapping exists.

## 1.5 Application of this research

This research is part of the development of CompetentNL<sup>5</sup>. CompetentNL is a Dutch national ontology for describing skills, competences and occupations. CompetentNL is necessary to create clear overview of skills and job descriptions in vacancies, profiles, study programs and courses in the Dutch labor market. Part of the project is the development of a method for semi-automatic ontology alignment, which is necessary for integration with other ontologies as described in Section 1.1. The team responsible for this aspect focuses on two parts, ontology matching and mapping refinement. This thesis represents the research on the mapping refinement. The goal is to incorporate both the developed ontology matching method, as well as the developed mapping refinement method in a tool, which will be used for creating various mappings between CompetentNL and other ontologies. The incorporation of the mapping refinement methods studied in this work is covered in Section 6.2 of the discussion.

## 1.6 Outline and main contributions of the thesis

This thesis is organized in the following manner. Initially, we examine the existing literature on ontology alignment and mapping refinement in Section 2, providing a scientific context for this study. Subsequently, we define the mapping refinement problem in Section 3. This is followed by the methods in section 4, where the methods used in the experiments

---

<sup>5</sup><https://www.competentnl.nl>

are explained. Next, the experimental setup is detailed in Section 5, outlining how the experiments were conducted and the results are shown. Finally, the thesis concludes with a discussion of the results in Section 6 and an overview of the conclusions and limitations and recommendations for future research in Section 7.

The main contributions of this thesis are the following. We give a comprehensive overview of how the use of domain knowledge evolved over time in the context of ontology alignment for both equivalence-only and extended relations. Secondly, we show that STROMA and the untrained version of TaSeR are not suitable to be used for mapping refinement in the labor market and biomedical domains. Third, we show that TaSeR works well, and we propose a simplified version of the TaSeR method, called S-TaSeR, to show that pre-fine-tuning has indeed a positive effect on the performance of mapping refinement via relation prediction with BERT. Additionally, we show that GPT-4 has good performance on the mapping refinement task, but that using domain knowledge still has the best performance. And lastly, we show that using domain knowledge as examples in a few-shot setting has a negative effect on the performance of GPT-4. In the conclusions in Section 7 we revisit these contributions and look at the implications and contributions for both practitioners as well as researchers.

## 1.7 Terminology

For the sake of clarity, we specify the main terms utilized in this thesis in this section. Note that these are not formal definitions, as these will be given in Section 3. We use the following terms:

- **Ontology alignment:** The process of finding correspondences between two different ontologies (source and target), where these correspondences may include a variety of relations, not just equivalence.
- **Ontology matching:** A type of ontology alignment, but focused only on finding correspondences with equivalence relations.
- **Correspondence:** A relation between two entities, one from the source and one from the target ontology.
- **Equivalence mapping:** A collection of correspondences resulting from ontology matching, having only equivalence relations.
- **Mapping refinement:** The process of refining an existing mapping by refining the existing equivalence relations into a set of extended relations such as is-a
- **Refined mapping:** A collection of correspondences with extended relations.
- **Domain knowledge:** Information that is related to the domain of the specific use case at hand. Examples are related ontologies. General sources of knowledge such as WordNet do not fall under domain knowledge.
- **Use case:** A scenario of ontology alignment where we have three data sources: the two ontologies and the existing mapping between them.

## Chapter 2

# Literature Analysis of Ontology Alignment with Extended Relations

In this section we will dive deeper into the literature on ontology alignment, mapping refinement and knowledge graph completion. In Section 1.2, we stated that the performance of two-step methods for creating a refined alignment is highly dependent on the ontology matching methods used in the first step. Therefore, we will start by providing a comprehensive overview of ontology matching in Section 2.1. Next, we will cover ontology matching with extended relations, i.e. one-step approaches, in Section 2.2.1. Afterwards in Section 2.2.2, we cover the refinement step in two-step approaches, i.e. mapping refinement. Finally, we will also include a short overview of knowledge graph completion in Section 2.3, as the mapping refinement techniques used in this thesis use ideas from that field of research.

### 2.1 Ontology matching

Ontology matching with equivalence relations has been extensively researched. The first mentions stem from 1986 [8]. Since then, a wide variety of methods have been developed.

#### 2.1.1 Classifying ontology matching methods

Various reviews and surveys have been written using different classification approaches [91, 109, 54]. One of the most recognized classifications is the one by Euzenat and Shvaiko [28]. They classified methods on the basis of the focus of the techniques used. The two major classes are element-level matching and structure-level matching.

In element-level matching, each entity within an ontology is considered isolated from the rest. Examples of element-level matching are: string-based, language-based and informal resource-based. String-based techniques look at the texts associated with entities as a collection of characters, without using the meaning of the words. Language-based techniques use natural language processing (NLP) methods to retrieve information from the meaning of words and sentence structure of the texts associated with entities. Informal resource-based techniques deduce relations between ontology entities based on how they are related to their associated resources such as links or pictures. Examples of ontology matching systems that use element-level techniques are GLUE, SAMBO and ASMOV [72, 63, 53].

The second group of techniques is structure-level matching. When matching an entity using structure-level matching, the connection to surrounding entities is used as extra information. Examples of structure-level techniques are graph-based and taxonomy-based techniques. Graph-based techniques view ontologies as labeled graphs, where the intuition is that if two entities are similar, their neighbors must also be similar. Taxonomy-based techniques are similar, with the difference being that they view the ontology as a taxonomy.

Ontology matching systems that use structure-level techniques include Similarity Flooding, OMEN and Cupid [75, 25, 73]. For a more comprehensive overview of which types of techniques are used in which method, we refer to the survey by Liu et al. [68].

### 2.1.2 Language-based techniques

Recent developments in ontology matching have shifted towards concentrating on element-level, language-based techniques [83, 81, 93]. These techniques, which are rooted in natural language processing, are broadly classified into two categories: intrinsic and extrinsic [28]. Intrinsic techniques analyze the content within the text, whereas extrinsic techniques use additional resources. Notably, thesauri such as WordNet [80] have long been integral to extrinsic methods in ontology matching. Systems like OLA [27], SAMBO [63], Falcon [49] and AgreementMaker [21] employ WordNet to interpret entities based on their specific meanings or contexts. This process often involves matching entities after determining their 'sense' as defined in WordNet. However, these extrinsic methods primarily leverage general external knowledge, with a notable absence of domain knowledge. This limitation arises from the constraints of thesauri and lexicons like WordNet, which offer limited scope and are not easily extendable. As a result, general knowledge sources have been preferred for their broader applicability in creating versatile tools. However, the introduction of word embeddings brought new opportunities to include domain knowledge.

### 2.1.3 Non-contextual word embeddings

Word embeddings are representations of natural language in a numeric format, often vectors. The idea of word embeddings has been around for a long time, with early proposals dating back to early nineties [47, 88, 95, 23]. Word embeddings have important advantages over other external information sources, such as improved understanding of language, over hand-created information sources. It has been shown that the resemblance between word embeddings is not just based on basic syntactic rules but also on semantic rules [78]. For instance, the calculation  $\text{embedding}(\text{"King"}) - \text{embedding}(\text{"Man"}) + \text{embedding}(\text{"Woman"})$  yields an embedding most similar to the one for "Queen" [78]. But most importantly, word embeddings can be trained on any text corpus and are therefore scalable and flexible. The aforementioned problem with thesauri and lexicons, that they can not easily be extended for the domain at hand and can therefore include domain knowledge, is not a problem for word embeddings as they can be trained using domain specific data. This opened the door for the use of domain knowledge. When word embeddings methods like word2vec [78] and fastText [11] had become widespread, they were also applied in ontology matching.

The first method to use of word embeddings in the context of ontology matching was proposed by Zhang et al. in 2014 [115]. They proposed a method combining Word2Vec embeddings trained on Wikipedia data with edit distance, a string metric. However, they did not yet use domain specific knowledge. After the publication of Zhang et al., similar works were published [108, 35]. The first mention of the use of domain specific knowledge was in the paper by Wang et al. [110]. They create a neural network that uses a word2vec embedding model which is trained using domain knowledge from the UMLS ontology. In their experiments they include ontology matching use cases from UMLS and SNOMED CT and show that their method achieves good performance on both. Another method that uses domain knowledge is the DeepAlignment [62]. They use a transfer learning approach to adapt pre-trained word embeddings to a given domain. Another method that uses domain knowledge is the Rafcom method [83]. They use a combination of word2vec embeddings and a random forest classifier and train it on domain knowledge from the ontologies of the use case. An example of the use word embeddings in combination with domain knowledge in the context of the labor market domain is the work by Giabelli et al. [30]. They looked into mapping the Italian labour ontology CP [1] to ESCO [2] using word embeddings. In their

approach they used the occupations from the ontologies, together with its description and all associated words such as synonyms to train a fastText embedding model, which was then used to align the two ontologies.

#### 2.1.4 Contextual word embeddings

While word embeddings as described above have beneficial properties, they also lack in certain areas. Such word embeddings are non-contextual, meaning a word always has the same meaning, independent of the context. This can be problematic, as words can be polysemic, meaning that they can have different meanings in different contexts. For example, the word "tear" will have the same embedding, whether it is used in "She shed a tear when she heard the news" or "Be careful not to tear the paper". While the word is spelled the same in each sentence, the meaning is different. Contextual word embeddings offer dynamic representations for polysemy, meaning that they recognize the context and assign different embeddings to "tear" based on its usage in a sentence. This use of context gives them a better understanding of language [48]. Another advantage of contextual word-embedding is the ability to represent out-of-vocabulary words. Since contextual word embeddings generate representations based on context, they can generate embeddings for words not seen during training, while non-contextual word embeddings are limited to the vocabulary it is trained on [69]. Publications using context-dependent embeddings demonstrate that these representations can attain top-tier results in various challenging NLP tasks [86, 24].

One of the first examples of the use of contextual word embeddings in ontology matching was the work by Neutel and de Boer [81]. They created a BERT-based method to create a mapping between the O\*NET and ESCO ontologies. However, they did not introduce domain knowledge into the model. The first application of contextual word embeddings in ontology alignment with domain knowledge was BERTMap [42]. BERTMap is an ontology matching system that utilizes a BERT model and fine-tunes it on the domain knowledge present in the ontologies of the use case. Guru Rao et al. [37] took on the same use case as Neutel and de Boer of aligning O\*NET and ESCO. In contrast to them, they did use domain knowledge. They fine-tuned a XLNet model [112], which was then used for ontology matching. Another approach is the Truveta Mapper [5]. It also uses BERT, but takes a different approach to incorporating the domain knowledge in the model. It defines the problem as a translation task, where it returns a path in the target ontology, based on the input entry. For training, the ontologies are transformed into paths which are used to fine-tune the BERT model.

#### 2.1.5 Generative large language models

Recently, new developments in the field of NLP have shifted from embedding-focused or encoder-based pre-trained language models towards decoder-based or generative large language models [55]. While models like BERT are built to encode textual information into a numerical format (text-to-embedding), models like GPT-4 are built for text-to-text tasks [3].

The first works that utilize generative large language models in ontology matching have recently been published. He et al. [43] show that generative large language models have the potential to outperform ontology alignment systems such as BERTMap given proper prompt engineering. Another method was proposed by Norouzi et al. [94]. They explore the usage of ChatGPT in ontology alignment. A different approach to using generative large language models is the proposed Olala method [44]. They use Sentence-BERT [92] to generate possible candidates. A generative large language model is then be used in a binary classification setting, outputting whether a candidate is correct or not, or in a multiple choice setting, in which the most likely entity is chosen from the different possibilities.

## 2.2 Ontology alignment with extended relations

All aforementioned papers on ontology matching focus on creating mappings where the relation of the established match is exclusively equivalence. Euzenat and Shvaiko noted that there were only a few methods that include other types of relationships than equivalence [28]. While it is often mentioned in papers that a mapping exists of correspondences between the source and target ontologies and a relation  $r$  such as *equivalence* or *is-a*, the relation is almost always set to equivalence [83, 42, 5]. Giunchiglia et al. [33] noted that other types of mapping relations can exist such as *is-a* or *part-of* relationships. While underexposed, there have been methods that include these relations, rather than exclusively the equivalence relation.

In Table 2.1 an overview is given of ontology matching and mapping refinement methods with extended relations. The *Steps* column indicates whether the method is a one-step approach (ontology matching) or a two-step approach (mapping refinement) and the *DK* column represents whether the method uses domain knowledge or not. Furthermore the types of relations identifiable by each of the methods is included. In Section 2.2.1 we will first look at the one-step methods and in Section 2.2.2 we look at the mapping refinement methods used in two-step methods.

Name	Steps	DK	Relation type						
			equivalence	is-a	part-of	related	disjoint	overlapping	close
S-Match [32]	1	no	x	x			x	x	
CtxMatch [14]	1	no	x	x					
AROMA [22]	1	no	x	x					
RiMoM [64]	1	no	x	x				x	
ASMOV [53]	1	no	x	x	x		x		
Taxomap [40]	1	no	x	x					x
Blooms [51]	1	no	x	x			x		
Blooms+ [52]	1	no	x	x			x		
CSR [101]	1	no		x					
STROMA [7]	2	no	x	x	x	x			
IUT [117]	1	no	x	x					
Dhouib et al. [104]	2	no		x					x
BERTsubs [17]	1	yes		x					
TaSeR [45]	2	yes	x	x	x	x	x		

Table 2.1: An overview of ontology alignment methods with extended relations

### 2.2.1 One-step approaches

The first one-step approaches were rule-based methods like CtxMatch and S-Match. Bouquet [14, 13] published CtxMatch in 2003. They made use of WordNet by translating the hierarchical relations between WordNet senses into a set of subclass relations in description logic, making them element-level methods with a focus on extrinsic language-based techniques. By matching the entities to the senses in the logic, relations could be determined. S-Match was later proposed by Kanade et al. [32] as a rationalised re-implementation of CtxMatch. The AROMA method [22] took a different, resource-based element-level matching, approach focusing on document-populated ontologies. It first creates entity hierarchies by creating association rules between entity names and terms in their documents. Afterwards it tries to find association rules between the hierarchies. Other rule-based methods include RiMoM [64], ASMOV [53], Taxomap [40], BLOOMS [51] and BLOOMS+ [52], CSR [101] and IUT [117].

Of the ontology matching methods that include extended relations, there were no methods that used either non-contextual or contextual word embeddings. The only exception is the contextual word embeddings method BERTsubs [17]. Not only did it use contextual word embeddings, it also included domain knowledge. The authors of the BERTmap method [42]

developed a method for predicting subclass relationships between classes of OWL ontologies using contextual semantic embeddings. They fine-tuned a BERT model using information from the source and target ontologies. Since the method uses both the textual information of the entities as well as the hierarchical structure, it is a combination of element and structure-level methods. The method not only works for finding subclass relations between two ontologies (inter-ontology) but also for finding missing subclass relations within an ontology (intra-ontology).

### 2.2.2 Two-step approaches (mapping refinement)

Compared to one-step methods, mapping refinement is an even smaller field on ontology alignment. To our knowledge only three methods exist that take an existing mapping and refine the relations from equivalence to extended relations.

The oldest method is the Semantic enrichment of Ontology Mappings (STROMA) method by Arnold and Rahm [7]. They include extended semantic relations on top of equivalence and subclass relations such as disjointness and overlap. STROMA is based on a three-step process: type computation, verification and selection. The type computation phase finds the type of semantic relation of a mapping using various linguistic strategies. The linguistic techniques include compounding, word sense disambiguation and background knowledge. The verification checks the predictions for potential missed equivalence relations. Next, the selection phase validates and filters the found relationships. Their method does not make use of domain knowledge. The details of this method will be explained in Section 4.1. The second method is the method proposed by Dhouib et al. [104]. They proposed an ontology mapping approach based on a set of rules exploiting the embedding space and measuring clusters of entity labels to discover the relationship between entities. They use fastText embeddings and do not use domain knowledge. They show that the combination of word embeddings and a measure of dispersion of the clusters of labels in the embedding space, makes it possible to determine not only equivalence relations but also subclass relations between entities. The most recent and current state-of-the-art method was developed in 2023 by Hertling and Paulheim. It is called Transformer-based Semantic Relation Typing (TaSeR) [45]. Like the BERTmap and BERTSubs models, they use a BERT model for classifying the relation between two given entities. Different from the BERTmap and BERTSubs models, they include a large number of relations and take an already existing mapping as input. The BERT model is pre-fine-tuned on a fine-tuned using domain knowledge from the source and target ontologies. Their method includes various semantic relationships such as equivalence, (inverse) is-a, (inverse) part-of, co-hyponym and disjointness. TaSeR is fine-tuned for mapping on a general data set and can be further fine-tuned for the test case at hand. The details of this method will be further explained in Section 4.2.2.1.

In mapping refinement we find the same pattern as for ontology matching: from rule-based to non-contextual word embeddings to contextual word embeddings. However, the recent rise of generative large language models has not yet been used in mapping refinement. There are works where generative large language models outperform domain specific models on mapping refinement related tasks such as document classification [18], sentiment analysis [29] and text classification [103, 31]. As these tasks are similar to mapping refinement, these findings spark the interest whether domain knowledge is still necessary for performing mapping refinement, or whether generative large language models are able to do this without a need for time-consuming pre-processing and fine-tuning of domain knowledge data.

## 2.3 Knowledge graph completion

A related field to mapping refinement is knowledge graph completion, in particular a subtask of knowledge graph completion (KGC): relation prediction. Relation prediction is very much

like mapping refinement, as the goal of both tasks is to determine the relation between two given entities. Before we dive into the similarities and differences between relation prediction and mapping refinement in Section 2.3.2, we will take a look at what the relation is between knowledge graphs and ontologies.

### 2.3.1 Relation between ontologies and knowledge graphs

Knowledge graphs, similar to ontologies and taxonomies, serve to model interconnected data. Since its inception and popularization by Google in 2012 [100], the exact definition has been a subject of debate. The initial introduction by Google did not provide a comprehensive definition, leading to diverse interpretations in subsequent years. In 2013, Blanco et al. described the construction of Yahoo’s knowledge graph through the alignment of new entities with an existing ontology [10]. They implicated that an ontology is a schema upon which a knowledge graph is built by adding instances. In contrast, Ehrlinger et al. did not agree with the notion of knowledge graphs as a novel technological advancement [26]. They critiqued the term’s usage as a buzzword and highlighted the confusion caused by its interchangeable use with ‘ontology’ and ‘knowledge base’. Later, the papers by Kejriwal et al. and Bonatti et al. in 2019 offered differing perspectives, with the former focusing on the fact-centric nature of knowledge graphs and the latter explaining them as a combination of a schema and facts [56, 12]. This definition of knowledge graphs, like Blanco’s, misses the fact that ontologies themselves can also have instances and are therefore not limited to serving as a schema for knowledge graphs. Meznar et al., in 2022, argued in the same way that knowledge graphs are synonymous with grounded ontologies [77]. Grounded meaning that the ontology is closely tied to real-world data and observations, thus containing instances. Based on these evolving definitions and interpretations, we conclude that knowledge graphs are essentially grounded ontologies. This similarity stems from their shared focus on structured, real-world data representation. Therefore, we can safely use techniques built for knowledge graphs in the context of ontologies.

### 2.3.2 Relation between relation prediction and mapping refinement

Now that we know how knowledge graphs and ontologies are related, we will look at KGC and in particular to relation prediction. Knowledge graphs, often created manually or semi-automatically, frequently miss numerous implicit entities and relationships, leading to a widespread issue of incompleteness. The goal of KGC is to solve the problems of incompleteness and sparsity caused by missing instances or links in knowledge graphs [19]. This technology enhances graph structures by predicting missing knowledge components like entities and relationships. KGC uses the RDF representation [60] of knowledge graphs existing of triples (*subject, object, predicate*) or as they are formulated in KGC: (*head, relation, tail*) [19]. This will be explained in more detail in section 3.1.2. Depending on the missing elements in the triples, KGC can be segmented into four tasks: tail prediction, head prediction, relation prediction and triple classification [19]. Tail prediction involves predicting a missing tail entity from the given head entity and relationship, for example: (*Data Scientist, works\_for, ?*). Head Prediction requires identifying a missing head entity based on the known relationship and tail entity, as in determining the person in (*?, leads, AI Research Team*). Relation prediction focuses on discovering the relationship between a known head and tail entity, for instance, figuring out the connection in (*Project Manager, ?, Software Development Team*). Lastly, triple classification entails evaluating the accuracy of a given triple, identifying if a triple such as (*Data Scientist, employed\_by, TechCorp*) is true.

Most KGC methods use knowledge graph embeddings, together with a scoring function [111, 65]. Examples of such methods are RESCAL [82], TransH [111], TransR [65] and ComplEx [105]. These specifically are not as much related with mapping refinement, as they assume a knowledge graph is fully connected, meaning that all entities are reachable from all

other entities in the knowledge graph. Since the goal of mapping refinement is to predict the relation between entities from two disconnected ontologies, it does not meet this assumption.

However, there are other approaches to KGC that do not rely solely on structural information, which are of use for mapping refinement. These methods use pre-trained language models like BERT. There have been multiple attempts to incorporate knowledge from knowledge graphs into pre-trained language models [67, 116, 87], but the first knowledge graph enhanced pre-trained language model specifically designed for KGC was presented in by Yao et al. [113]. Their method, named KG-BERT, converts each triple into a textual sentence, utilizing the textual information of its head entity, relation and tail entity. Subsequently, it approaches KGC as a text classification task, where BERT is fine-tuned using triples from the knowledge graph during the fine-tuning process. The domain knowledge methods in this thesis make use of the idea introduced in the KG-BERT method. In Section 4.2.1.6 we explain the details of how KGC with KG-BERT works.

## Chapter 3

# Defining the Mapping Refinement Problem

In this section, we will formalize the mapping refinement problem. First, we will define what an ontology is. Next, we will review the two tasks of ontology alignment, ontology matching and mapping refinement, and define them.

### 3.1 Ontologies

In this thesis, we cover ontologies, which are sets of terms and definitions specific to a certain field. These definitions can be simple or detailed, making ontologies range from basic term lists to entity-relationship models and logics [28]. We focus on a specific type of ontologies, called hierarchical ontologies or taxonomies. A hierarchical ontology can be represented as a directed acyclic graph in which a node and its label represent a concept and its meaning. In this thesis we will use the label of a concept to refer to the concept itself. Relationships among the nodes are usually represented by *is-a* or *inverse is-a* relations in the graph [58]. A hierarchical ontology is not rich in terms of data type properties, relationships among concepts and axioms [97, 39].

#### 3.1.1 Defining ontologies

Having established an understanding of what an ontology is, we will use this section to give a formal definition. We will use a simplified version of the formal definition of an ontology as defined by Euzenat and Shvaiko [28]. We describe an ontology as a tuple  $\langle C, R, = \rangle$ , such that:

- $C$  is the set of classes;
- $R$  is the set of relations;
- $=$  is a relation over  $C \times R \times C$  called assignment.

Not all classes need to have all relations. Relations can be directional but do not have to be. In Figure 3.1 a subset of the ESCO ontology is given. On the left all variables are assigned, on the right that same ontology is visualised.

Variable	Example
$C$	System analysts, ICT consultant, ICT advisor, Computer vision engineer, Computer vision specialist
$R$	synonym, is-a
$=$	ICT consultant = is-a = System analysts, Computer vision engineer = is-a = System analysts, ICT consultant = synonym = ICT advisor, Computer vision engineer = synonym = Computer vision specialist

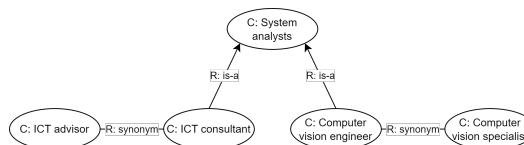


Figure 3.1: Example of a subset of the ESCO ontology

### 3.1.2 Ontology languages

Now that we have formally defined what ontologies are, we will look at which language to use to model ontologies according to our formal definition. There are multiple languages to describe an ontology. The most widely used are OWL and RDF. Web Ontology Language (OWL) [9] is a comprehensive language used for creating complex ontologies on the semantic web. It enables detailed description of classes, properties and their interrelationships, supporting advanced features like logical inference. Resource Description Framework (RDF) [60] is a standard model for data interchange on the web, primarily used to represent information about resources and their interrelationships. It employs a simple triple structure for representing semantic data. As the ontologies we want to describe are not complex and there is no need to model constraints or logical inferences, we will use the the RDF language. The triple structure of the RDF language looks like (*subject*, *predicate*, *object*). *Subject* refers to the resource being described. It can be anything with an identity, typically represented by an uniform resource identifier (URI). The *predicate* indicates the type of relationship that exists between the *subject* and the *object*. The *object* is the value or resource to which the *subject* is related via the *predicate*. In this thesis we will use the RDF notation used in knowledge graph completion (KGC) where they use head, relation, tail instead of subject, predicate, object. The notation of a triple becomes  $(h, r, t)$ . As an example, the ontology from Figure 3.1 can be described in RDF with KGC notation as: (*ICT consultant*, *is-a*, *System analysts*), (*Computer vision engineer*, *is-a*, *System analysts*), (*ICT consultant*, *synonym*, *ICT advisor*), (*Computer vision engineer*, *synonym*, *Computer vision specialist*). The language used for the mappings will be SKOS [79]. SKOS is a lightweight vocabulary designed for representing simple knowledge structures. It provides a way to represent, among others, mappings. It is built upon RDF, allowing concepts to be easily integrated with other data on the semantic web.

## 3.2 Ontology alignment

After defining ontologies and the language for their description, we will look at the definition of ontology alignment. The goal of ontology alignment is to create a collection of correspondences  $\{(h_1, r_1, t_1), (h_2, r_2, t_2), \dots, (h_n, r_n, t_n)\}$ , called a mapping. Where  $h$  is an entity from the source ontology,  $t$  is an entity from the target ontology and  $r$  represents the relation that holds between the two entities. As previously discussed, alignment can be approached in two ways: a one-step process and a two-step process. This thesis will concentrate on the latter as it is more general and flexible as highlighted in Section 1.2 of the introduction. Two-step ontology alignment consists of ontology matching followed by mapping refinement. The process of ontology matching is performed by using an ontology matcher, which gives an equivalence mapping as output. This mapping only contains correspondences with equivalence relations, which are then refined by a mapping refiner into a refined mapping. For a visual overview we refer to Figure 1.3 in the introduction. We will start by defining ontology matching, after which we define mapping refinement.

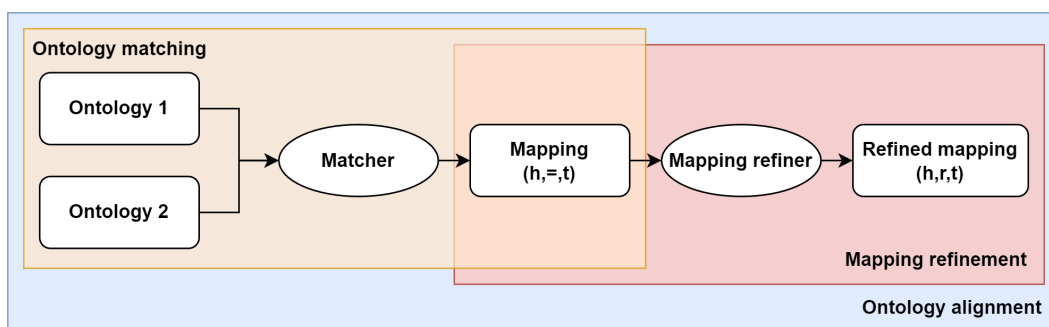


Figure 3.2: A visualisation of two-step ontology alignment

### 3.2.1 Definition of ontology matching

As described in the previous section, we need an ontology matcher in order to match ontologies. We define a matcher according to the definition of Euzenat and Shvaiko [28]. A matcher can be defined as a function, denoted as  $f(O1, O2, M', p, b) = M$ . In this formulation,  $M$  represents the resulting mapping.  $O1$  and  $O2$  represent the source and target ontologies.  $M'$  represents a pre-existing mapping which could be used as background knowledge. This variable could potentially be an empty set. The variable  $p$  encompasses the additional parameters integral to the matching process, while  $b$  signifies the external background knowledge sources that are employed in the process.

Having gained an understanding of what a matcher is, we can define how it is used in ontology matching. To recap, the fundamental challenge involves establishing a mapping  $M$  between two distinct ontologies,  $O1$  and  $O2$ . By applying the matcher, we get the resulting mapping. This mapping is defined as  $\{(h_1, =, t_1), (h_2, =, t_2), \dots, (h_n, =, t_n)\}$ , where  $h$  represents an entity from the source ontology,  $t$  represents an entity from the target ontology and  $=$  represents the equivalence relation that holds between them.

### 3.2.2 Definition of mapping refinement

Now that we have created a mapping with equivalence relations using a matcher, we can refine this mapping into the final refined mapping. We define the mapping refiner in a similar manner as the matcher functions from Euzenat and Shvaiko [28]. The difference being that we include an existing equivalence mapping. We define a mapping refiner as the function:  $f(O1, O2, M^=, p, b) = M$  where  $O1$  and  $O2$  are the source and target ontologies,  $M^=$  is the mapping with only equivalence relations that we want to refine,  $p$  the additional parameters and  $b$  the external background knowledge sources. Using the mapping refiner we refine a mapping  $\{(h_1, =, t_1), (h_2, =, t_2), \dots, (h_n, =, t_n)\}$  into a mapping  $\{(h_1, r_1, t_1), (h_2, r_2, t_2), \dots, (h_n, r_n, t_n)\}$ . For this thesis, we assume the matching phase has already been executed, leaving us with a mapping with exclusively equivalence relations that needs to be refined. In the next section we will describe which relations  $r$  can be.

### 3.2.3 Semantic relationship types of a refined mapping

With our understanding of all elements of mapping refinement established, we will look at the relations the refiner has to be able to identify. In Table 3.1 the semantic relationship types used in this thesis are given. Two words are synonyms if they have the same or nearly the same meaning in some or all contexts. For example, "database administrator" and "database manager" are synonyms because they can be used interchangeably in many contexts to describe someone who is responsible for a database. A hyponym is a word whose meaning is included within the meaning of another, more general word. The relationship is one of

inclusion. For instance, "database administrator" is a hyponym of "computer occupations" because every database administrator is someone who has a computer related occupation, but not all people who have computer related occupations are database administrators. So if we have a triple (*database administrator*, *skos:broadMatch*, *computer occupations*), this means that the term database administrator is more narrowly defined than computer occupations, which is more general. A hypernym is the opposite of a hyponym; it is a word with a broad meaning that more specific words fall under. A hypernym is a kind of umbrella term. For instance, "manager" is a hypernym of "retail manager" because it encompasses retail managers along with other types of managers. So if we have a triple (*manager*, *skos : narrowMatch*, *retail manager*), this means that the term manager is more broadly defined than retail manager, which is less general. To describe these relations, we will use the SKOS language. We will leave out the prefix skos: in the rest of this thesis.

<b>Linguistic relation</b>	<b>Example</b>	<b>Set theory</b>	<b>SKOS</b>
Synonymy	ICT consultant, ICT advisor	Equivalence	skos:exactMatch
Hyponymy	ICT consultant, System analysts	Subset-of	skos:broadMatch
Hypernymy	System analysts, ICT consultant	Inclusion, Superset-of	skos:narrowMatch

Table 3.1: Correspondence between linguistic relations and SKOS concepts

# Chapter 4

## Methods

In this section we will explain the methods used for mapping refinement. We start by explaining the STROMA method. Following this, we delve into the domain knowledge methods. As both TaSeR and our proposed S-TaSeR method use BERT, we will first explain the specifics of BERT models. After having explained BERT, we will explain the TaSeR and S-TaSeR methods.

### 4.1 STROMA

STROMA is a rule-based method. The STROMA method can be expressed according to the mapping refiner definition from Section 3.2.2 as  $f(\emptyset, \emptyset, M^-, \emptyset, b) = M$  where  $O1$  and  $O2$  are empty,  $M^-$  is the mapping we want to refine,  $p$  is empty and  $b$  is the background knowledge present in WordNet, OpenThesaurus and UMLS Metathesaurus. The method exists of three steps: type computation through five distinct strategies, type verification and selection. The method is visualised in Figure 4.1 and the three steps will be further explained in the next sections.

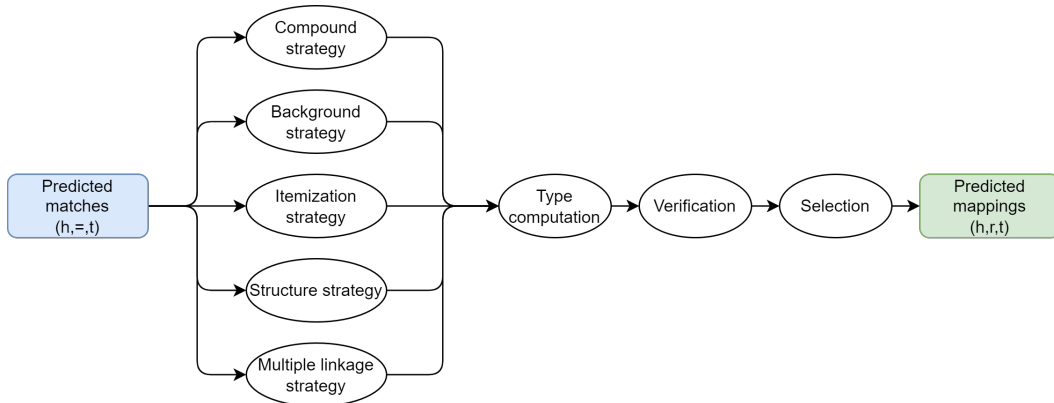


Figure 4.1: A visualisation of the STROMA method

#### 4.1.1 Type computation strategies

STROMA employs five strategies to determine relation types, each providing a confidence score or marking as undecided for each correspondence. These scores are combined using a weighted scoring system based on experimental reliability, with the highest-scoring relationship type chosen. In cases where all strategies are undecided, *equivalence* is the default

prediction. For ties, the preference order is: *equivalence*, *is-a*, *inverse is-a*, *has-a*, *part-of* and *related*. The five strategies will be explained in the next sections.

#### 4.1.1.1 Compound strategy

The compound strategy identifies "is-a" relationships by analyzing word compounds, where a compound (*t*) ending with another entity (*h*) suggests *t* is a type of *h*. It requires *t* to be at least three characters longer than *h* to avoid pseudocompounds and uses a list of common prefixes to exclude prefix derivations. A limitation of this strategy is that it struggles with evolved spellings or unique morphemes not in dictionaries. It has a confidence weight of 1.0.

#### 4.1.1.2 Background knowledge strategy

The background knowledge strategy overcomes lexical limitations in mapping refinement by using external knowledge sources like WordNet, OpenThesaurus and UMLS Metathesaurus. It leverages these sources to identify semantic relations (e.g., synonyms, hypernyms, hyponyms) for finding correspondences beyond mere lexical similarities. A key method is Gradual Modifier Removal, where modifiers are sequentially removed from compounds to check for matches in a background dictionary, determining the semantic relation type. Its effectiveness is limited by the dictionaries' scope, often missing new or domain-specific terms. This strategy has a confidence weight of 1.0.

#### 4.1.1.3 Itemization strategy

The third strategy, the itemization strategy is applied when at least one entity in a correspondence is an itemization, defined as a list of items without commas, slashes, or conjunctions. The process involves simplifying item sets by removing synonyms, hyponyms and hypernyms within and between the sets. The remaining items in the sets are compared to determine the relationship type: *equal*, *is-a*, *inverse is-a*, or *undecided*. The strategy is particularly useful in scenarios where one entity encompasses multiple items or categories. The itemization strategy has a confidence weight of 1.0.

#### 4.1.1.4 Structure strategy

The structure strategy uses ontologies' explicit structures to infer semantic relationships, analyzing the connection between an entity and its parent to deduce relationships with other entities. For example, if entity Y is related to parent X and a known relationship exists between X and Z, this information suggests a relationship between Y and Z. It incorporates WordNet and compounding for relationship determination, excelling in scenarios where direct semantic links are not obvious but inferable from the ontology structure. This strategy carries a confidence weight of 0.8.

#### 4.1.1.5 Multiple linkage strategy

The multiple linkage strategy identifies hierarchical relationships by focusing on schema elements with multiple correspondences, suggesting that an entity linked to several others may represent a more general category. This approach has the limitation that it depends on the accuracy of initial matches, being vulnerable to false positives that can result in more false positives. It has a confidence weight of 0.5.

### 4.1.2 Verification step

After determining the relationship type of a correspondence using the five strategies, a verification step is implemented to address subtleties and confirm or revoke the initial

decision. This step is crucial because *is-a* correspondences can be misleading due to different hierarchical organizations in the input ontologies. For instance, consider the correspondence (“apparel.children\_shoes”, “clothing.children.shoes”). Initially, both compound and background knowledge strategies might suggest an "is-a" relationship, as "children\_shoes" obviously fall under "shoes." However, a detailed examination of the hierarchical paths can reveal that these entities are, in fact, equal. The verification step involves combining the leaf entity with its parent entity and comparing this combination to the unaltered leaf entity of the corresponding pair. If this combined entity matches the opposite leaf entity, the previously assumed *is-a* relation is reconsidered. For example, combining "children" and "shoes" from one ontology and finding it equivalent to "children\_shoes" in another may lead to an equivalence determination, revoking the initial *is-a* relation.

### 4.1.3 Selection step

In the selection step, the focus is on filtering out less likely correspondences, especially those without a clear semantic relation identified in the enrichment phase. The process is streamlined with two main rules. The first rule makes sure correspondences with high confidence are directly accepted into the final mapping. The second rule makes sure that correspondences with intermediate confidence are included only if they are supported by at least one semantic relation strategy. This step is designed to enhance the final mapping by considering correspondences that may not have been identified in the original matcher setting, including those that are approximately equivalent or represent different types of relations. However, since the focus of the thesis is on a scenario with completed matching and an existing mapping, the details of this selection process are less important.

## 4.2 Domain knowledge methods

In this section we will explain the methods that use domain knowledge: TaSeR and S-TaSeR. Since both methods use BERT, we will first explain the details of this model.

### 4.2.1 BERT

Bidirectional Encoder Representations from Transformers (BERT), developed by Devlin and colleagues at Google in 2019 [24], is one of the most widely used pre-trained contextual language models, utilizing a multi-layered bidirectional Transformer encoder, based on the original transformer model, from Vaswani et al.’s work [107]. As opposed to directional models such as RNNs, which read the text input sequentially (left-to-right or right-to-left), the transformer encoder reads the entire sequence of words at once, so in both directions. This characteristic allows the model to learn the context of a word based on all of its surroundings, both left and right of the word. This lead to the name of contextual word embeddings.

#### 4.2.1.1 BERT input tokenization

Before texts can be processed by the model, they need to be tokenized. Tokenization entails transforming the input text into a list of machine readable tokens. In a BERT tokenizer are three types of tokens: word tokens, [CLS] tokens and [SEP] tokens. The word tokens are the main tokens representing the words in the text. BERT utilizes wordpiece tokenization, which breaks down words into smaller units (subwords or characters) if they are not in the vocabulary. For instance, the word "sleeping" is split up into subwords "sleep" and "###ing", where "###" indicate a continuation or part of a larger word. Each word token has its own numerical representation. The second type of token is the [CLS] token, which is added at the beginning of each input sequence. When BERT is trained on classification

tasks, the final hidden state (the transformer output) corresponding to this [CLS] token is used as the aggregate representation of the entire sequence. Lastly, the [SEP] token is used to clearly delineate the end of one sentence and the start of another. This helps BERT understand sentence boundaries within the input. We will cover an example of what the tokenized input looks like in Section 4.2.1.6.

After translating the input text into token embeddings and adding the special tokens, two additional embedding types are generated: segment embeddings and positional embeddings. Segment embeddings indicate for each subword token to which of the [SEP] separated segments of the input they belong. Positional embeddings indicate the position of the subword token in the whole input text. As mentioned before, BERT is a bidirectional model instead of a directional one. Unidirectional models have the advantage that the positional information is inherent in the input format. The positional embeddings make sure that our bidirectional model can incorporate the positional information just as well. The word, segment and positional embeddings are combined into a single vector through element-wise addition. An example of the tokenized input is visualised in Figure 4.2

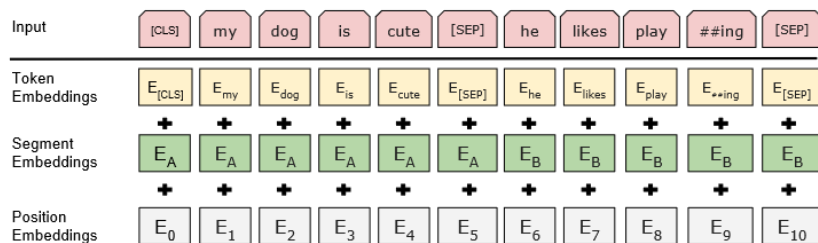


Figure 4.2: BERT’s tokenizer combines token embeddings, segmentation embeddings and position embeddings into a single input vector. Image from Devlin et al. [24].

#### 4.2.1.2 BERT model architecture

After tokenization the inputs can be used as input for the model. BERT’s architecture is a subset of the original transformer implementation in Vaswani et al. [107]. Different from the original transformer model, which had the goal of translation through encoding and decoding, the goal of BERT is to create an embedding through encoding only. Therefore it does not include the decoder layers and only uses the encoder layers. The encoder consists of multiples of pairs of multi-head attention and feed-forward networks, called transformer blocks. A visualisation of this can be found in Figure 4.3. Like Vaswani et al.’s transformer, the number of transformer blocks used in BERT can vary. BERT base has twelve transformer blocks, while BERT large has twenty four blocks. Increasing the number of transformer blocks enhances the model’s capability to understand language, but it also linearly raises the computational complexity. For a more detailed description of the architecture and mechanics of transformer models we refer to the original paper on transformers by Vaswani et al. [107].

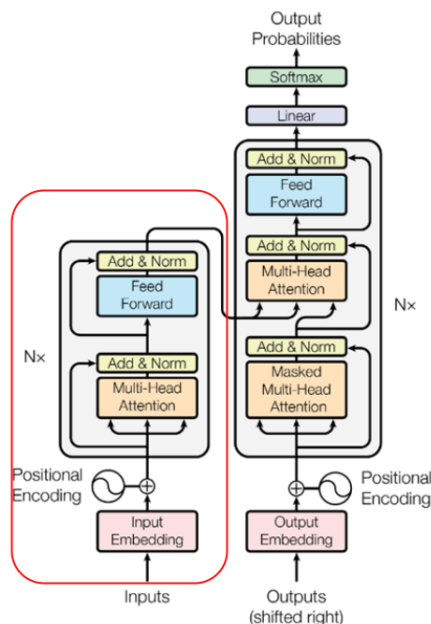


Figure 4.3: The original transformer architecture, with the BERT architecture highlighted by the red rectangle. Original image from Vaswani et al. [107]

#### 4.2.1.3 Pre-training BERT

Having explored BERT's tokenization process and architecture, we will next delve into the model's training methodology. Training BERT exists of two phases: pre-training and fine-tuning. In the pre-training phase, a huge corpus of unlabeled text is fed into the model using an unsupervised learning approach. The original BERT model has been trained on the English Wikipedia corpus and the BooksCorpus data set. The English Wikipedia dataset consists of approximately 2,500 million words, while the BooksCorpus dataset contains approximately 800 million words of English text from a variety of published books. During pre-training, the data from the corpora is fed to the model using two tasks: Masked Language Modeling (MLM) and Next Sentence Prediction (NSP). The goal of MLM is to learn the model a deep bidirectional representation of sequence context. In MLM, a random percentage of the input tokens are masked and the objective of the model is to predict the original meaning of these masked tokens, given the context provided by the other, non-masked, tokens in the sequence. Specifically, 15% of the words are masked in each sequence. For example, in the sentence "The quick brown fox jumps over the lazy dog", the model might replace the word "brown" by "[MASK]" and predict its original meaning during training. The goal of NSP is to enhance BERT with an understanding of the relationship between two sentences. NSP is a binary classification task, where the model is fed pairs of sentences and must predict if the second sentence is the consequent of the first sentence in the original document. In 50% of the cases, the second sentence is the actual following sentence; in the other 50%, it is a random sentence from the corpus. After performing both MLM and NSP, the model can be used for various tasks such as tokenization and embedding generation, feature extraction for texts and classification tasks.

#### 4.2.1.4 Fine-tuning BERT

While directly using a pre-trained BERT model for a task is possible, the model is often fine-tuned further for a specific task since this almost always enhances the performance [24]. For fine-tuning, we load a pre-trained model and continue training on all parameters

using labeled data for specific supervised learning tasks, like sentence pair classification or question answering. Depending on the task, additional classification layers can be added to the model. Unlike pre-training, where the model learns general features of a language, fine-tuning adjusts the model’s parameters to specialize in the nuances and requirements of the specific task. A smaller learning rate is typically used compared to pre-training. This is because we want to make smaller, more precise updates to the weights, avoiding overwriting the language understanding learned during pre-training. After fine-tuning, the model can be used for the task at hand.

#### 4.2.1.5 The DistilBERT variant

The development of BERT has led to several variants, each with unique characteristics and optimizations. In this thesis we will focus on the DistilBert variant [96]. Initial experiments with other variants, the original BERT [24] and RoBERTa [70], revealed that DistilBERT offers comparable performance to these variants but with reduced computational time. Therefore we decided to exclude the other BERT variants from this study. Furthermore, the TaSeR method, which is central to our research, uses the DistilBERT variant. Therefore, focusing on DistilBERT aligns with our objective to validate the TaSeR method’s effectiveness, making the examination of other BERT variants less relevant to our research question.

DistilBERT is a compact and efficient adaptation of the original BERT model, distinguished primarily by its reduced size and faster training capabilities. The efficiency of DistilBERT comes from a technique called knowledge distillation, where DistilBERT learns to replicate the original BERT’s probability predictions. Unlike BERT, DistilBERT does not perform the Next Sentence Prediction (NSP) task; instead, it concentrates on closely mirroring the original model’s predictions for the Masked Language Modeling (MLM) tasks and ensuring its hidden state representations maintain high similarity with those of BERT. As a result, DistilBERT provides a resource-friendly option, achieving substantial performance levels comparable to the original BERT with less computational complexity.

#### 4.2.1.6 Knowledge Graph Completion Using BERT

One of the tasks BERT can be fine-tuned for is knowledge graph completion (KGC). BERT’s application in knowledge graph completion was introduced in the KG-BERT method [113]. The KG-BERT method fine-tunes a BERT model using knowledge graphs in RDF triple format. Within the notion of knowledge graph completion, multiple subtasks exist as explained in Section 2.3.2. In this thesis we focus on the relation prediction task. The objective of relation prediction is to determine the relationship  $r$  between a head and a tail entity, expressed as  $(h, t) = r$ . This task is similar to sentence pair classification, given its structure in classifying two sentences, in this case the entities [113]. First, the input data is tokenized as described in Section 4.2.1.1. As an example, consider the triple (database administrator, broadMatch, computer occupations). The entities *database administrator* and *computer occupations* are tokenized as  $[[CLS], database, administrator, [SEP], computer, occupations, [SEP]]$ . Relations are encoded using a label encoder, which replaces every label with an integer. This is done by first sorting the labels alphabetically. Our labels *broadMatch*, *exactMatch* and *narrowMatch* become 0, 1 and 2 respectively. In the case of our example, the *broadMatch* relation is thus transformed into 0. All triples in the training dataset are processed this way and the tokenized data is used for fine-tuning. Upon completion, the model is tested with a similarly tokenized test set. The model outputs the predicted relations as integer labels, which are translated back into relation labels using the label encoder. As relation prediction and mapping refinement are very much related as explained in Section 2.3.2, we can use this form of fine-tuning and prediction in mapping refinement.

### 4.2.2 Mapping refinement methods that use BERT

Now that we have explained the details of the BERT models and how it can be used for mapping refinement, we will examine the mapping refinement methods that use these models. We will first explain the details of the TaSeR method, after which we will go into detail about our proposed simplified version of TaSeR, called S-TaSeR.

#### 4.2.2.1 TaSeR

The first mapping refinement method that uses domain knowledge we cover is the TaSeR method. The TaSeR method uses the DistilBERT model in a similar fashion as the KGBERT method. The TaSeR method exists of two components: pre-fine-tuning on an external knowledge source in triple format and fine-tuning on use case data. The method is set up in such a way that the method is already ready for use after the pre-fine-tuning and can be fine-tuned further if domain knowledge is available for the use case at hand. In the next sections we will explain both the pre-fine-tuned variant, which we will call TaSeR base, and the fine-tuned variant, which we will call TaSeR.

**TaSeR base** The TaSeR base method can be expressed according to the mapping refiner definition from Section 3.2.2 as  $f(\emptyset, \emptyset, M^=, p, b) = M$  where  $O1$  and  $O2$  are empty,  $M^=$  is the mapping we want to refine,  $p$  is the list of hyperparameters of DistilBERT, which have not been published by the authors, and  $b$  is the language knowledge present in DistilBERT and the WDS dataset existing of the external knowledge sources WordNet, DBpedia and Schema.org. The goal of the use of the WDS dataset is to incorporate knowledge about semantic relations in the model. The original paper states that Wikidata is also during pre-fine-tuning. However, according to the authors, the final version of the model is trained on the WDS dataset only. In the classification task used during pre-fine-tuning, a total of 7 classes is used: *equivalence*, *is-a*, *inverse is-a*, *part-of*, *inverse part-of*, *cohyponym* and *no match*. To avoid confusion: a cohyponym is a word that is one of multiple hyponyms of another word. In Table 4.1 you find the distribution of relations in the WDS dataset.

Ontology	Relationship type					Total
	equivalence	(inv.)is-a	(inv.)part-of	cohyponym	negatives	
Wordnet	215,672	84,501	9,092	44,329	410,960	764,554
DBpedia	58	246	0	198	0	502
Schema.org	0	1,421	0	826	0	2,247

Table 4.1: Distribution of relations in the ontologies that make up the WDS dataset

The process of the method is as follows. The first step is to fine-tune the DistilBERT model using the triples from DBpedia, Wordnet and Schema.org. After having fine-tuned the model, it can directly be used to predict the relations of a given mapping to output a refined mapping. The TaSeR base model was uploaded to Hugging Face<sup>1</sup>. The hyperparameters used during pre-fine-tuning have not been published. In Figure 4.4 the TaSeR base method is visualised.

<sup>1</sup><https://huggingface.co/dwsunimannheim/TaSeR>

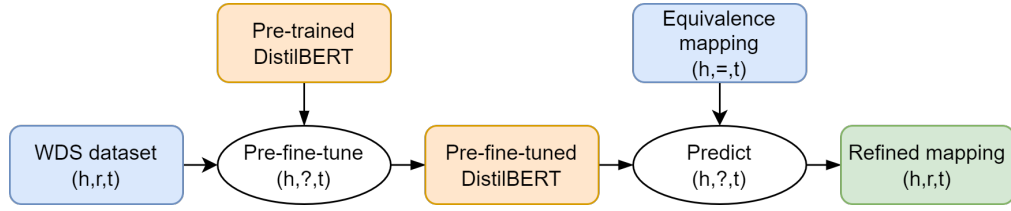


Figure 4.4: A visualisation of the TaSeR base method

**Fine-tuned TaSeR** This version further fine-tunes the TaSeR base model on specific use case data. The pre-fine-tuned model is used and further fine-tuned on the RDF triples from the domain ontologies. The fine-tuned TaSeR method can be expressed according to the mapping refiner definition from Section 3.2.2 as  $f(O1, O2, M^=, p, b) = M$  where  $O1$  and  $O2$  are the source and target ontologies,  $M^=$  is the equivalence mapping we want to refine,  $p$  is the list of hyperparameters of DistilBERT, specified later in this section and  $b$  is the language knowledge present in DistilBERT and the external knowledge in the WDS dataset.

The process exists of two phases. In the first phase, the DistilBERT model is pre-fine-tuned as described in the previous section. Next, the source and target ontologies are converted into RDF triples. These are used to fine-tune the pre-fine-tuned DistilBERT model. The resulting fine-tuned model can be used to refine a mapping into a refined mapping. A visualisation of the fine-tuned TaSeR method can be found in Figure 4.5.

In the original paper on TaSeR [45], Arnold and Rahm employed hyperparameter tuning in the development, using the test set for validation purposes. Recognizing this as data leakage, we have decided not to use this approach. We did conduct a small experiment to see if it would be possible with a different approach. Our experiments involved splitting the training dataset into separate training and validation segments, where the validation segment was used as validation in hyperparameter tuning. However, these experiments yielded suboptimal results. The discrepancy between the characteristics of the training and test sets led to hyperparameters that performed well on the validation set but poorly on the test set. Consequently, we have chosen not to use hyperparameter tuning in the main experiments and instead utilize a predetermined set of hyperparameters. We use a max token length of 100, 3 epochs and a learning rate of  $1 \times 10^{-5}$ . In order to limit the runtime of the fine-tuning, we let the batch size depend on the size of the datasets. For the web, labor market and biomedical use cases we use a batch size of 8, 64 and 128 respectively. The use cases will be covered in Section 5.1.2.

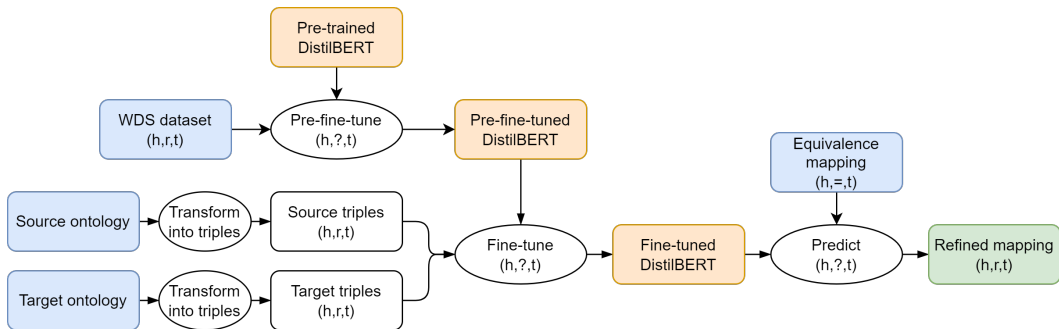


Figure 4.5: A visualisation of the TaSeR method

#### 4.2.2.2 S-TaSeR

The idea of pre-fine-tuning as performed in the TaSeR method is unconventional and made us wonder whether the pre-fine-tuning step actually had any effect on the performance

of DistilBERT. Therefore we propose a simplified version of the TaSeR method, called S-TaSeR, where the main difference is in the absence of the pre-fine-tuning step. The proposed method is shown in Figure 4.6. The method can be implemented multiple BERT models, but as mentioned before, we will focus on the DistilBERT variant. The S-TaSeR method can be expressed according to the mapping refiner definition from Section 3.2.2 as  $f(O1, O2, M^=, p, b) = M$  where  $O1$  and  $O2$  are the source and target ontologies,  $M^=$  is the mapping with only equivalence relations we want to refine,  $p$  are the hyperparameters of DistilBERT, specified later in this section and  $b$  is the language knowledge present in DistilBERT.

The process is as follows. First, the source and target ontologies are converted into RDF triples. These are used to fine-tune the DistilBERT model. We use the distilbert-base-uncased model from Hugging Face<sup>2</sup>. The resulting fine-tuned model can be used to refine a mapping into a refined mapping. A visualisation of the S-TaSeR method can be seen in Figure 4.6. For fine-tuning we use a max token length of 100, 3 epochs and a learning rate of  $1 \times 10^{-5}$ . In order to limit the runtime of the fine-tuning, we let the batch size depend on the size of the datasets. For the web, labor market and biomedical use cases we use a batch size of 8, 64 and 128 respectively. The use cases will be covered in Section 5.1.2.

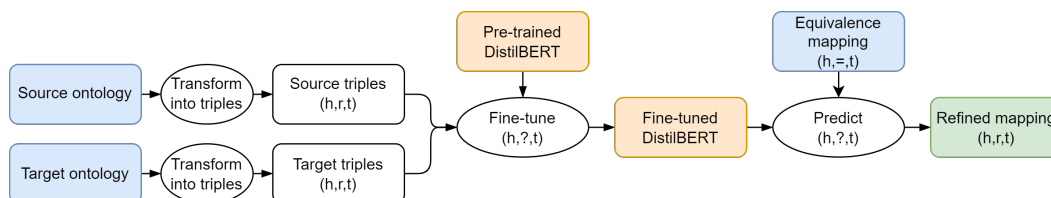


Figure 4.6: A visualisation of the S-TaSeR method

### 4.3 The GPT-4 method

Now that we have a clear image on the methods that use domain knowledge, we will review the last method. This method utilizes the GPT-4 model. First we will explain what GPT-4 is and afterwards we will describe how we use it to perform mapping refinement.

#### 4.3.1 GPT-4

First we will delve into GPT-4, a state-of-the-art generative large language model developed by OpenAI [3]. Notably, while OpenAI has previously released both the weights and technical details of GPT2 [90] and the technical details (but not the weights) of GPT3 [15], the organization has opted not to disclose either for GPT-4.

GPT-4 is grounded in the transformer architecture, as outlined in Vaswani et al.’s paper [107]. It was pre-trained to predict subsequent tokens in a text sequence. The training corpus comprised a blend of publicly accessible internet data and datasets procured from third-party sources. However, details on these sources are not available. Post pre-training, GPT-4 was further refined via Reinforcement Learning from Human Feedback (RLHF), as described in Christiano et al. [20].

Our research primarily leverages GPT-4’s zero-shot and few-shot learning capabilities. In the zero-shot scenario, the model tackles tasks it has not encountered during its training phase, without any preliminary examples. It uses its pre-trained knowledge and language understanding to generate responses. For instance, when tasked with categorizing a news article as sports, politics, or technology, GPT-4 assesses and classifies based on its text comprehension, without prior examples.

<sup>2</sup><https://huggingface.co/distilbert/distilbert-base-uncased>

The second approach, few-shot learning, entails providing GPT-4 with a limited set of examples that serve as a contextual or stylistic guide for the task at hand. Continuing with our news categorization example, this approach would involve supplying the model with a few articles and their corresponding labels, thereby aiding its classification process.

From a practical point of view, GPT-4 can be used via an API connection. We use the "gpt-4-1106-preview" model <sup>3</sup>. The model can be used by "calling" it using two prompts: a system message and a user message. System messages are instructions on the broad task that you are using the model for. The user message is the specific example of that broad task. Besides the messages, the system takes one other input: the temperature. The temperature is a parameter that controls the randomness of the responses generated. The temperature ranges from 0.0 to 2.0. A lower temperature results in more deterministic and predictable responses, while a higher temperature leads to more variability and creativity in responses. In the experimental evaluation we will experiment with various temperatures to see which one works best for our task. In the next section we will explain how we use GPT-4 for mapping refinement.

### 4.3.2 Mapping refinement with GPT-4

Mapping refinement using GPT-4 is performed by combining the two entities we want to know the relation of into a single prompt, which is fed into the model. In the next sections we will explain the two options of using GPT-4: zero-shot and few-shot.

#### 4.3.2.1 GPT-4 zero-shot

The GPT-4 zero-shot (ZS) mapping refinement method can be expressed according to the mapping refiner definition from Section 3.2.2 as  $f(\emptyset, \emptyset, M^{\leftarrow}, p, b) = M$  where  $O1$  and  $O2$  are empty,  $M^{\leftarrow}$  is the equivalence mapping we want to refine,  $p$  is the temperature and  $b$  is the external knowledge present in GPT-4. The process begins by loading the mapping with only equivalence relations and creating a prompt for each correspondence. This prompt is then used to run the model. We have experimented with various prompts and constructed the following prompt based on the best outcomes. The system message used is: *"You are a relation prediction expert. You know the relation between two given concepts. The choices of relations are exactMatch, narrowMatch and broadMatch. Only output the predicted relation without any extra words, characters, or symbols."* We experimented with mentioning "skos:" before the relations, but this had a negative effect on the performance. The user message used is: *"Determine the relation between the concepts: [concept1], and: [concept2]"*, where *concept1* and *concept2* are the two entities. After feeding the prompt into the model, it returns the predicted relation, which is used in the refined mapping. We looked into ways the method could be sped up. An option would be to group multiple samples into one prompt. However, we decided not to use this approach, to ensure independence of the predicted relations. Grouping them into one prompt could possibly created correlation between the group and the prediction. In Figure 4.7 the method is visualised.

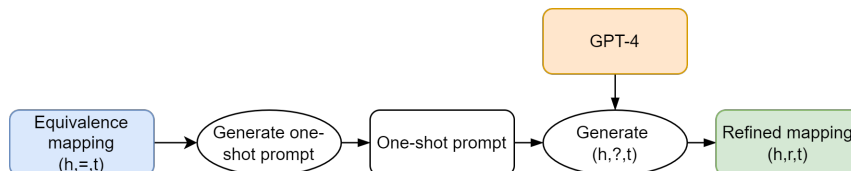


Figure 4.7: A visualisation of the GPT-4 zero-shot method

<sup>3</sup><https://platform.openai.com/docs/models/gpt-4-and-gpt-4-turbo>

#### 4.3.2.2 GPT-4 few-shot

The GPT-4 few-shot (FS) mapping refinement method can be expressed according to the mapping refiner definition from Section 3.2.2 as  $f(O1, O2, M^=, p, b) = M$  where  $O1$  and  $O2$  are the source and target ontologies,  $M^=$  is the mapping with only equivalence relations we want to refine,  $p$  is the temperature and  $b$  is the external knowledge present in GPT-4.

The method starts with loading the mapping with only equivalence relations. For each correspondence in the mapping, the entity to which it is connected to in its ontology via the *broadMatch* relation is added to the set of examples. The inverse of the *broadMatch* triple, a *narrowMatch* triple, is also added to the set of examples. Lastly the *exactMatch* triples of both the head and the tail are added to the example set. These examples are used as few-shot examples in the user prompt. The system message is specified as: *"You are a relation prediction expert. You know the relation between two given concepts. The choices of relations are exactmatch, narrowmatch and broadmatch. Only output the predicted relation without any extra words, characters, or symbols."* The user message contains both the triples from the set of example as well as the two entities we want to predict the relation for. For each triple from the set of examples we generate this sentence: *"Relation between concepts: [entity1], and: [entity2] is [relation]\n"*, where  $\backslash n$  makes sure each example triple is on a new line. This text with example string is combined with the following sentence after which it is used for prompting: *"Determine the relation between the concepts: [concept1], and: [concept2]"*. The model returns the predicted relation, which is used in the refined mapping. In Figure 4.8 the method is visualised.

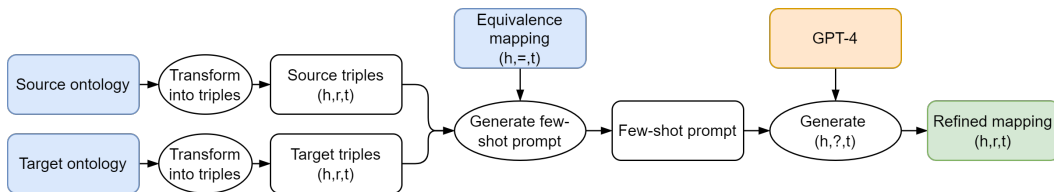


Figure 4.8: A visualisation of the GPT-4 few-shot method

## Chapter 5

# Empirical Evaluation

### 5.1 Experimental methodology

In this section we will explain how the experiments are set up. First, we will look at how the experiments will be structured. We will cover the use cases included, how they are pre-processed and the evaluation methods used. Then we will cover the results of the experiments.

#### 5.1.1 Experiments

In this section we will explain how we set up the experiments to answer the research questions.

##### 5.1.1.1 Experiment 1: Comparing the current state-of-the-art methods that do not use domain knowledge to those that do

In order to answer the first subquestion: *"How do current mapping refinement methods that incorporate domain knowledge compare to methods that do not?"*, we will compare the current state-of-the-art methods. The goal of this experiment is to find out what the current differences in performance are between methods that do not use domain knowledge and those that do. We will compare methods that do not use domain knowledge, STROMA and base TaSeR, to a method that does, TaSeR with fine-tuning. When using BERT models, it is important to conduct multiple runs due to inherent variabilities such as initialization sensitivity, data shuffling effects and overall model variance. Multiple runs mitigate the risks of underestimating or overestimating its effectiveness based on a singular potentially atypical outcome. By aggregating results from several iterations, we can derive more reliable results, thereby giving a more accurate representation of the model's generalizability and stability. Therefore we will run both the TaSeR method 5 times for each use case. The TaSeR base method does not have this problem, as it is not fine-tuned in our experiments.

##### 5.1.1.2 Experiment 2: Validating TaSeR

To answer the second subquestion: *"Does including additional general knowledge via pre-fine-tuning, as done in the current state of the art of mapping refinement, benefit its performance?"*, we will compare the performance of TaSeR to S-TaSeR. As mentioned before, TaSeR is pre-fine-tuned on a general data set of triples with various semantic relationships. The goal of this experiment is to find out whether this pre-fine-tuning is beneficial for the performance of the setup. The results of TaSeR from the first experiment will be used here. S-TaSeR will be evaluated similarly to TaSeR in the first experiment, by running the method 5 times for each use case. The mean of the macro average F1 scores will be used.

### 5.1.1.3 Experiment 3: Comparing methods that use domain knowledge and GPT-4 in a zero-shot setting

To answer the third subquestion: *"How do state-of-the-art mapping refinement methods compare to generative large language models in a zero-shot setting?"*, we will compare the TaSeR and S-TaSeR methods to GPT-4 in zero-shot setting (ZS). The goal of this experiment is to find out whether the hassle of incorporating domain knowledge is still necessary to effectively perform mapping refinement. The experiment is split up into two parts: temperature tuning and the main experiment.

In the temperature tuning experiment we will experiment with various temperature settings of GPT-4 on the web use cases, as these are relatively small, to find out which has the best performance. In order to account for the randomness in GPT-4 responses, we run the experiment 3 times and we take the mean of the resulting scores.

In the main part of the experiment, we compare the TaSeR and S-TaSeR methods against the GPT-4 ZS method with the best performing temperature. We run the GPT-4 model in a zero-shot setting and use the results of TaSeR and S-TaSeR from the first and second experiment for comparison. While GPT-4 has the same problem as the BERT-based methods of randomness in its predictions, we will not run the method multiple times due to cost limits.

### 5.1.1.4 Experiment 4: Comparing GPT-4 in a zero-shot and a few-shot setting

The last experiment is set up to answer the last subquestion: *"Does incorporating domain knowledge into generative large language models through few-shot learning enhance their performance in mapping refinement?"*. We will use GPT-4 for mapping refinement in both a zero-shot and a few-shot (FS) setting. The goal of this experiment is to find out whether generative large language models like GPT-4 can benefit from domain knowledge, or that it might not be worth the hassle to incorporate it. Like for the previous experiments, we will use the existing results of GPT-4 ZS method in the zero-shot setting.

## 5.1.2 Use cases

We will use a total of eleven datasets from three domains: web, labor market and biomedical. The overview of the number of relations per ontology and mapping is given in Table 5.1. W, L and B represent the use case categories Web, Labor market and Biomedical respectively.

Use case	Dataset	Relationship type		Total
		equivalence	(inv.)is-a	
Diseases (W1)	Yahoo	0	1,082	1,082
	DMOZ	0	5,121	5,121
	Mapping	314	36	350
Furniture (W2)	Amazon	0	173	173
	Ebay	0	26	26
	Mapping	12	99	111
Groceries (W3)	Amazon	0	57	57
	Ebay	0	332	332
	Mapping	28	126	154
Clothing (W4)	Amazon	0	29	29
	Ebay	0	151	151
	Mapping	6	214	220
ESCO-CompetentNL (L1)	ESCO	2,789	3,577	6,366
	CompetentNL	0	4,322	4,322
	Mapping	989	4,486	5,475
ESCO-O*NET (L2)	ESCO	2,789	3,577	6,366
	O*NET	0	1,312	1,312
	Mapping	498	2,280	2,778
DOID-NCIT (B1)	DOID	3,084	9,388	12,472
	NCIT	15,762	9,359	25,121
	Mapping	3,280	4,667	7,947
OMIM-ORDO (B2)	OMIM	9,120	4,300	13,420
	ORDO	5,700	12,164	17,864
	Mapping	2,602	143	2,745
FMA-SNOMED CT body (B3)	FMA	32,765	88,931	121,696
	SNOMED CT	14,148	13,459	27,607
	Mapping	5,077	7,699	12,776
NCIT-SNOMED CT neoplasm (B4)	NCIT	20,247	9,956	30,203
	SNOMED CT	9,517	2,693	12,210
	Mapping	2,663	298	2,961
NCIT-SNOMED CT pharm (B5)	NCIT	22,136	22,109	44,245
	SNOMED CT	2,772	3,527	6,299
	Mapping	4,062	5,913	9,975

Table 5.1: An overview of the content of the use cases

### 5.1.2.1 Web use cases

The web use cases were created for evaluating the performance of the STROMA method [6]. The datasets were later also used for evaluating the TaSeR method. The mapping used as the test set was filtered to only contain correspondences where both entities are present in the ontologies. Furthermore, they filtered out various small errors in both the ontologies as well as the mappings. The web use cases used in the TaSeR paper exists of seven use cases. We will not use three of them, as one of them is in German, one of them does not have any subclass relations in its ontologies and one of them has a source ontology without subclass relations. The hierarchical relations in the ontologies are represented by *rdfs:subClassOf*. The ontologies in the web use cases do not contain any synonym-like relations.

**Diseases (W1)** The first dataset is the Diseases use case, an extract from a disease mapping project that aligns terminologies between directories of Yahoo directory service<sup>1</sup> and the Directory of Open Access Journals (DMOZ)<sup>2</sup>. Yahoo operated a directory service that categorized websites, including those related to health and diseases. These categories were manually curated and gave a structured approach to classifying websites. The DMOZ, was a multilingual open-content directory of World Wide Web links. DMOZ was known for its extensive volunteer editor base, which could lead to a diverse range of categories and subcategories based on different criteria, potentially differing from Yahoo’s approach.

<sup>1</sup><https://web.archive.org/web/20141122194515/https://dir.yahoo.com/>

<sup>2</sup><http://www.dmoz.org/>

**Furniture (W2), Groceries (W3) and Clothing (W4)** The other three use cases all focus on ontologies from amazon.com<sup>3</sup> and ebay.com<sup>4</sup>. The first one, Furniture (W2), is a mapping between the furniture categories. The second one, Groceries (W3), is a mapping between categories of groceries and food & beverages. And the third amazon-ebay mapping, Clothing (W4) is between clothing categories. Amazon categories are typically much more specific than Ebay categories, so that the Amazon ontologies are generally more comprehensive than the Ebay ontologies. To keep the development and mapping effort at a moderate level, only the top two category levels of the Amazon directory were regarded. In Table 5.1 you find the distribution of relations of the two ontologies and the mapping. It is important to note that differences between the datasets used in the original TaSeR paper and those we use are caused by the removal of duplicates in our dataset. The web use cases are relatively small and have rough differences between the entities in the ontologies.

### 5.1.2.2 Labor market use cases

The second category of use cases are the labor market use cases. The mappings have been created as a part of the development of ESCO. The European Commission encourages member countries to use ESCO. If a country creates a national labor ontology, the European Commission has the duty to create and maintain a mapping between ESCO and the country's national labor market ontology as established in 2018 in Regulation (EU) 2016/589(33)<sup>5</sup>. Therefore, all country members that have a national labor market ontology have a mapping available through the ESCO website. While the use cases from the other two categories have been created with the goal of evaluating equivalence and subclass ontology alignment, the use cases from the labor market domain have not. Therefore they are a good test set to test the performance of ontology alignment methods in a real-world scenario. However, as the goal is not to be a stable data set for evaluation, they are more likely to contain human errors and inconsistencies. For the ontologies in the labor market domain, hierarchical relations are represented by *skos:broader*. Only the ESCO ontology includes some form of synonyms, being the relations *altLabel*.

**ESCO-CompetentNL (L1)** The first of the two labor market use cases is the ESCO-CompetentNL use case. ESCO, developed by the European Commission, is an ontology of European skills, competences, qualifications and occupations. It serves as a reference tool for education, training, job searching and workforce development in the European Union. ESCO classifies occupations and skills/competences in a hierarchical structure and links them to qualifications and jobs across Europe. CompetentNL is an ontology currently under development tailored to the Dutch labor market. It is not yet publicly available. The used versions of the ontologies are version 1.0.3 for ESCO and version 0.91 for CompetentNL. The original data for the ESCO-CompetentNL use case is in Dutch. Since the underlying models for TaSeR and S-TaSeR are the english variants of BERT and STROMA is also only available for English (and German), we decided to translate the data to English in order to be able to get meaningful results. We use the DeepL.com translator<sup>6</sup> to translate the ontologies from Dutch to English. In order to confirm the correctness of the translation, we have checked 5% of the translated data by hand. In this check we did not find big mistakes or errors, indicating that the translation was performed correctly. ESCO contains the predicate *altLabel*, which can be used to create *exactMatch* triples. CompetentNL does not contain any synonyms.

<sup>3</sup>[www.amazon.com](http://www.amazon.com)

<sup>4</sup>[www.ebay.com](http://www.ebay.com)

<sup>5</sup>[https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=uriserv%3A0J.L\\_.2016.107.01.0001.01.ENG](https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=uriserv%3A0J.L_.2016.107.01.0001.01.ENG)

ENG

<sup>6</sup><https://www.deepl.com/nl/translator>

**ESCO-O\*NET (L2)** Our second labor market domain also uses ESCO and aligns it to O\*NET. The same version of ESCO is used as for L1. O\*NET, developed for the U.S. Department of Labor, is an ontology that contains detailed descriptions of the labor market including information on skills, abilities, knowledge, work activities and interests associated with the occupations. As mentioned before, ESCO has alternative labels. O\*NET does not have any type of synonym relations.

### 5.1.2.3 Biomedical use cases

These use cases are from the Bio-ML track of the OAEI<sup>7</sup>, created by He et al. [41]. They address the limitations of the OAEI in evaluating ML-based systems. They introduce five new biomedical OM use cases involving ontologies from Mondo[106] and UMLS[50]. These use cases include both equivalence and subclass matching and offer a comprehensive evaluation framework for both ML-based and non-ML-based OM systems. The biomedical use cases can be described as very domain specific and have relatively subtle differences between entities. The subclass relation present in all ontologies is *rdfs:subClassOf*. The synonym relations vary per ontology and will be described in the use case specific section below.

**NCIT-DOID (B1)** By combining the National Cancer Institute Thesaurus (NCIT)<sup>8</sup> with the Human Disease Ontology (DOID)<sup>9</sup>, this dataset targets cancer and general disease ontologies. The National Cancer Institute Thesaurus (NCIT) offers a detailed set of terms covering various aspects of cancer and biomedical research. DOID (Human Disease Ontology) provides a standardized hierarchy of human disease terms, promoting consistency in annotations across databases. Besides subclass relations, NCIT and DOID have synonym relations in the form of *Thesaurus:P90* and *oboInOwl:hasExactSynonym* respectively.

**OMIM-ORDO (B2)** This dataset integrates Online Mendelian Inheritance in Man (OMIM)<sup>10</sup>, a detailed database of human genes and genetic disorders, with Orphanet Rare Disease Ontology (ORDO)<sup>11</sup>, an ontology focusing on rare diseases. OMIM is a comprehensive, authoritative compendium of human genes and genetic phenotypes. It provides detailed information about genetic disorders and their relationships to genes. ORDO focuses on rare diseases, offering a structured vocabulary for rare disease representation, facilitating data interoperability in clinical and research settings. Besides subclass relations, OMIM and ORDO have synonym relations in the form of *skos:exactMatch* and *efo:alternative\_term* respectively.

**SNOMED-FMA Body (B3)** Combination of Systematized Nomenclature of Medicine Clinical Terms (SNOMED CT)<sup>12</sup> and the Foundational Model of Anatomy (FMA)<sup>13</sup>, with a focus on anatomical structures. SNOMED CT (Systematized Nomenclature of Medicine Clinical Terms) is a comprehensive, multilingual clinical healthcare terminology. The Foundational Model of Anatomy (FMA) is an ontology of the structures and relationships in the human body, beneficial for bioinformatics and computational biology. Besides subclass relations, SNOMED CT and FMA have synonym relations in the form of *skos:altLabel* and *fma:synonym* respectively.

<sup>7</sup><https://oaei.ontologymatching.org/>

<sup>8</sup><https://ncit.nci.nih.gov/ncitbrowser/>

<sup>9</sup><https://disease-ontology.org/>

<sup>10</sup><https://www.omim.org/>

<sup>11</sup><https://www.orpha.net/consor/cgi-bin/index.php#:~:text=The%20Orphanet%20Rare%20Disease%20ontology,usable%20data%20for%20computational%20analysis.>

<sup>12</sup><https://www.snomed.org/>

<sup>13</sup><https://www.nlm.nih.gov/research/umls/sourcereleasedocs/current/FMA/index.html>

**SNOMED-NCIT Neoplas (B4)** Focused on neoplastic processes, it combines SNOMED CT with NCIT. The synonym relations of SNOMED CT and NCIT are the same as in the other use cases that include the same ontologies.

**SNOMED-NCIT Pharm (B5)** This dataset brings together pharmacologic substances from SNOMED CT and NCIT. The synonym relations of SNOMED CT and NCIT are the same as in the other use cases that include the same ontologies.

### 5.1.3 Data preparation

In this section we will explain how the ontologies will be transformed into train datasets and how the mappings are transformed into test data sets.

#### 5.1.3.1 Train sets

The ontologies are translated into a set of samples in RDF KGC triple format with a head, relation and tail:  $\{(h_1, r_1, t_1), (h_2, r_2, t_2), \dots (h_n, r_n, t_n)\}$ . The two ontologies in a use case are transformed into triples and combined into one train set. For generating the *broadMatch* relation we use the *rdfs:subClassOf* or *skos:broader* relation. The *narrowMatch* triples are generated by using the inverse relation between *broadMatch* and *narrowMatch*. For example, taking the inverse of the triple  $(headcook, broadMatch, cook)$  results in the triple  $(cook, narrowMatch, headcook)$ . The *exactMatch* triples are generated by using the synonym-like relations in each ontology. Additionally, for every entity in each ontology a triple where the head and tail refer to the same entity is included, called a self-loop triple. For instance, the self-loop triple for the entity *cook* is  $(cook, exactMatch, cook)$ . This addition ensures that training datasets for use cases where ontologies lack any form of synonym-like relations, also incorporate *exactMatch* triples. After having generated all triples, we filter out *broadMatch* and *narrowMatch* triples of which the head and tail entities are also present in an *exactMatch* triple. We do this since the ontologies sometimes include hierarchical relations like  $(cook, narrowMatch, cook)$ . As these can cause ambiguity in the train set, these triples are dropped. The distribution of relations in the resulting train sets can be seen in Table 5.2.

Note that B4 has a very large difference between the number of *exactMatch* triples and the number of *broadMatch* and *narrowMatch* triples. This is caused by the fact that many entities in the NCIT neoplas ontology contain a large number of synonym (P90) labels.

Use case	Relationship type			Total
	exactMatch	broadMatch	narrowMatch	
Diseases (W1)	3,153	4,504	4,506	12,163
Furniture (W2)	170	192	192	554
Groceries (W3)	360	386	386	1132
Clothing (W4)	121	154	154	429
ESCO-CompetentNL (L1)	7,899	7,376	7,375	22,650
ESCO-O*NET (L2)	4,889	4,889	4,888	14,666
NCIT-DOID (B1)	18,787	18,389	18,390	55,566
OMIM-ORDO (B2)	18,787	18,389	18,390	49,363
FMA-SNOMED CT body (B3)	102,390	102,390	102,388	307,168
NCIT-SNOMED CT neoplasm (B4)	29,764	12,649	12,658	55,071
NCIT-SNOMED CT pharm (B5)	25,636	25,636	25,642	76,914

Table 5.2: An overview of the distribution of relations in the train datasets

#### 5.1.3.2 Test sets

We format the test sets in the same manner as the train sets:  $\{(h_1, r_1, t_1), (h_2, r_2, t_2), \dots (h_n, r_n, t_n)\}$ . Regarding the web use cases, the mapping of W4 does not contain any *broadMatch* triples. In order to make sure all use cases are a three class classification problem, we generate the

inverse of the *broadMatch* triples as *narrowMatch* triples in the same manner as for the train sets described in Section 5.1.3.1. For the labor market use cases, the mappings also include more types of relations such as *skos:related*, but for these triples were removed as they are not the focus of the research. The mappings of the biomedical use cases are given as pairs in two separate tsv files, one containing the equivalence correspondences and one containing the subclass correspondences. The subclass correspondences are all in one direction, equal to *broadMatch*. We generate the inverse of the *broadMatch* triples as *narrowMatch* triples in the same manner as for the train sets described in Section 5.1.3.1. In order to save costs on GPT-4 API calls and reduce runtime, we set a maximum of 5000 samples for the biomedical test sets using random undersampling. The test sets after processing can be seen in Table 5.3.

Use case	Relationship type			Total
	exactMatch	broadMatch	narrowMatch	
Diseases (W1)	314	25	11	350
Furniture (W2)	12	95	4	111
Groceries (W3)	28	14	112	154
Clothing (W4)	6	107	107	134
ESCO-CompetentNL (L1)	989	1957	2529	5475
ESCO-O*NET (L2)	498	2053	227	2778
NCIT-DOID (B1)	2062	1469	1469	4998
OMIM-ORDO (B2)	2602	72	71	2745
FMA-SNOMED CT body (B3)	1985	1507	1507	4999
NCIT-SNOMED CT neoplasm (B4)	2663	149	149	2961
NCIT-SNOMED CT pharm (B5)	2036	1482	1482	5000

Table 5.3: An overview of the distribution of relations in the test datasets

#### 5.1.4 Evaluation Methods

In this thesis, we adopt a comprehensive approach to evaluate the performance of the proposed methods by leveraging multiple evaluation methods. Specifically, we employ macro-average F1 scores, confusion matrices and boxplots. This section provides an overview of these methods.

##### 5.1.4.1 Macro-average F1 score

The Macro-average F1 score is a robust metric for evaluating model performance across multiple classes, particularly useful in imbalanced datasets. As most of the test sets are imbalanced, we have chosen this metric over accuracy. The macro-average F1 score exists of the precision and recall for each class combined in the F1 score, which is averaged over all classes. The relevant equations are as follows:

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

$$F1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

$$\text{Macro-average F1} = \frac{1}{N} \sum_{i=1}^N F1_i$$

where  $N$  is the number of classes and  $F1_i$  is the F1 score for class  $i$ .

### 5.1.4.2 Boxplot

Boxplots provide a graphical representation of the distribution of a set of values. We include them to visualise the variance of macro-average F1 scores. A boxplot illustrates the median of the data, the interquartile range (IQR) and outliers. This visualization helps in identifying variability between method performances.

### 5.1.4.3 Confusion Matrix

A confusion matrix offers a detailed breakdown of a model’s predictions, showing the correct and incorrect predictions across different classes. It is a square matrix, with each row representing the instances in a predicted class and each column representing the instances in an actual class. This evaluation method is included to investigate the performance of the included methods on individual classes.

## 5.2 Experimental results

In this section, we present the results of our experiments. Although we applied all methods to all use cases and could show them in a single large table, we chose to display only the relevant results for each experiment for clarity. The complete table is available in the appendix.

### 5.2.1 Experiment 1: Comparing the current state-of-the-art methods that do not use domain knowledge to those that do

The results of the first experiment are shown in Table 5.4. The F1 scores show that STROMA has decent performance on the web use cases. However, on the other use cases it does not have good performance, with no better than random results. TaSeR base has worse results, with no better than random scores for any of the use cases. The results show that TaSeR has the best performance. Note that TaSeR has close to random performance for B3. Examining the runtime for each method reveals that both STROMA and TaSeR base complete within one minute for all use cases, whereas TaSeR requires more time, extending up to 27.5 minutes for the B3 use case.

	Random	STROMA	TaSeR base	TaSeR
W1	0.33 (-)	0.33 (0.1)	0.21 (0.4)	<b>0.44</b> (2.2)
W2	0.33 (-)	0.41 (0.1)	0.29 (0.4)	<b>0.46</b> (0.5)
W3	0.33 (-)	0.39 (0.1)	0.25 (0.4)	<b>0.80</b> (0.6)
W4	0.33 (-)	0.56 (0.1)	0.30 (0.5)	<b>0.83</b> (0.6)
L1	0.33 (-)	0.11 (0.9)	0.23 (0.5)	<b>0.48</b> (2.6)
L2	0.33 (-)	0.18 (0.5)	0.31 (0.4)	<b>0.45</b> (1.8)
B1	0.33 (-)	0.24 (0.9)	0.29 (0.5)	<b>0.90</b> (5.3)
B2	0.33 (-)	0.07 (0.5)	0.10 (0.4)	<b>0.58</b> (4.7)
B3	<b>0.33</b> (-)	0.00 (0.9)	0.17 (0.6)	<b>0.34</b> (27.5)
B4	0.33 (-)	0.02 (0.5)	0.09 (0.4)	<b>0.82</b> (5.3)
B5	0.33 (-)	0.00 (0.9)	0.26 (0.5)	<b>0.85</b> (7.3)

Table 5.4: Mean macro average F1 score for TaSeR and macro average F1 score for Random classification, STROMA and TaSeR base

#### 5.2.1.1 Investigating individual runs for experiment 1

We focus on the B3 case to find out why the TaSeR score is so low. In order to do this we zoom in on one of the five runs of TaSeR for that use case. We pick the run with the median score. We visualise the predictions in Table 5.5. In the confusion matrix we find that the TaSeR method has a tendency to predict *exactMatch* for most correspondences in the test

set. The precision for the *broadMatch* and *narrowMatch* triples is high, as the model does not have any false positives for those relations

		Predicted			Classification Report		
		broadMatch	exactMatch	narrowMatch	Precision	Recall	F1-Score
Actual	broadMatch	235	1272	0	1.00	0.16	0.27
	exactMatch	0	1985	0	0.42	1.00	0.60
	narrowMatch	0	1419	88	1.00	0.06	0.11
					Macro avg. F1		0.33

Table 5.5: Combined confusion matrix and classification report for the median run of the TaSeR runs on use case B3

### 5.2.2 Experiment 2: Validating TaSeR

The results of the second experiment shown in Table 5.6. The scores show that TaSeR generally performs best, with S-TaSeR achieving the same scores for the W1, L1, B1, B2 and B3 use cases. For the other use cases S-TaSeR is following TaSeR closely, except for W3 and W4, where the difference is larger. For the B3 use cases both methods score just above random performance. Upon examining the runtimes of the methods, it is observed that the runtimes for TaSeR with fine-tuning and S-TaSeR are very similar, albeit S-TaSeR requires a marginally longer duration overall.

	Random	TaSeR	S-TaSeR
W1	0.33 (-)	<b>0.44</b> (2.2)	<b>0.43</b> (2.2)
W2	0.33 (-)	<b>0.46</b> (0.5)	0.39 (0.5)
W3	0.33 (-)	<b>0.80</b> (0.6)	0.62 (0.6)
W4	0.33 (-)	<b>0.83</b> (0.6)	0.47 (0.5)
L1	0.33 (-)	<b>0.48</b> (2.6)	<b>0.48</b> (2.7)
L2	0.33 (-)	<b>0.45</b> (1.8)	0.40 (1.9)
B1	0.33 (-)	<b>0.90</b> (5.3)	<b>0.90</b> (5.6)
B2	0.33 (-)	<b>0.58</b> (4.7)	<b>0.58</b> (5.0)
B3	0.33 (-)	<b>0.34</b> (27.5)	<b>0.35</b> (29.4)
B4	0.33 (-)	<b>0.82</b> (5.3)	0.79 (5.5)
B5	0.33 (-)	<b>0.85</b> (7.3)	0.82 (7.6)

Table 5.6: Macro average F1 score for Random, TaSeR and S-TaSeR

In Figure 5.1 the distribution of all individual runs is visualised. On the y-axis you find the macro average F1 score and on the x-axis you find the use cases. Each box represents one of the two methods for the use case on the x-axis. For the web use cases we see that TaSeR is more constant in its predictions as the boxes are compact and there are little outliers. For S-TaSeR we see that the scores are lower and have a bigger spread, particularly for the W4 dataset. If we look at the labor market use cases, we find that the methods are more similar in their predictions. Both methods are relatively constant and the scores are close. For the first of the biomedical use cases, we find that both methods have high scores and low variance. The scores for the second use case B2 are lower and S-TaSeR has a higher variance. For B3 both methods have low scores and higher variance. The result for B4 show higher scores and low variance for both methods, with S-TaSeR’s median being lower than that of TaSeR. For the last use case, we find that TaSeR has a higher score and lower variance than S-TaSeR. Note that S-TaSeR has a higher maximum score than TaSeR for the B2 and B3 use cases.

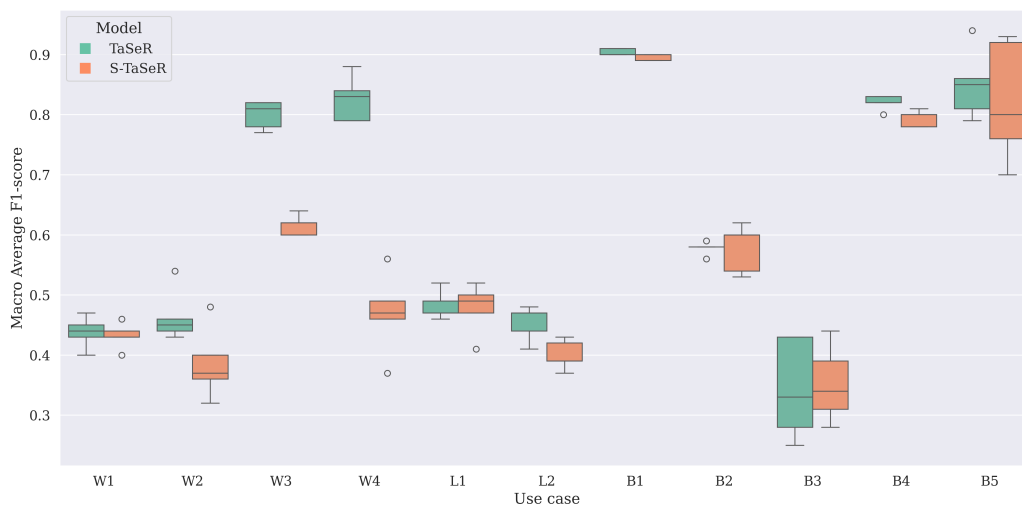


Figure 5.1: Macro average F1 score for TaSeR and S-TaSeR method per use case

### 5.2.2.1 Investigating individual runs for experiment 2

To investigate the large difference in performance between the two methods on the W4 use case, we also visualise the predictions of the median run for those methods in tables 5.7 and 5.8. While TaSeR has decent scores across all relations, we find that S-TaSeR cannot efficiently differentiate between the *narrowMatch* and *broadMatch* relations. S-TaSeR classifies almost all triples with *broadMatch* relations as *narrowMatch*.

		Predicted			Classification Report			
		broadMatch	exactMatch	narrowMatch	Precision	Recall	F1-Score	
Actual	broadMatch	8	2	97	0.73	0.07	0.14	
	exactMatch	0	4	2	0.57	0.67	0.62	
	narrowMatch	3	1	103	0.51	0.96	0.67	
					Macro avg. F1			0.47

Table 5.7: Combined confusion matrix and classification report for the median run of the S-TaSeR runs on use case W4

		Predicted			Classification Report			
		broadMatch	exactMatch	narrowMatch	Precision	Recall	F1-Score	
Actual	broadMatch	104	1	2	0.87	0.97	0.92	
	exactMatch	2	4	0	0.67	0.67	0.67	
	narrowMatch	14	1	92	0.98	0.86	0.92	
					Macro avg. F1			0.83

Table 5.8: Combined confusion matrix and classification report for the median run of the TaSeR runs on use case W4

In the last experiment we saw that TaSeR had a tendency to over-predict the *exactMatch* relation for use case B3. To check whether the low score of S-TaSeR for that use case has the same characteristics, we show the confusion matrix and classification report for the median run in Table 5.9. Similarly to TaSeR, we find that S-TaSeR has a low score, as it overpredicts the *exactMatch* relation.

		Predicted			Classification Report		
		broadMatch	exactMatch	narrowMatch	Precision	Recall	F1-Score
Actual	broadMatch	13	1494	0	1.00	0.01	0.02
	exactMatch	0	1985	0	0.43	1.00	0.60
	narrowMatch	0	1129	378	1.00	0.25	0.40
					Macro avg. F1		0.34

Table 5.9: Combined confusion matrix and classification report for the median run of the S-TaSeR runs on use case B3

### 5.2.3 Experiment 3: Comparing methods that use domain knowledge and GPT-4 method in a zero-shot setting

In this section we will first show the results of the temperature tuning experiment, after which we will look at the results of the main experiment where we compare the results of TaSeR and S-TaSeR with GPT-4.

#### 5.2.3.1 Experiment 3.1: Temperature tuning GPT-4

In Table 5.10 the macro average F1 scores for the temperature tuning experiment are shown. The scores for W1 are the highest at temperature 0.00. For W2 the performance is the highest for temperatures 0.25 and 0.50. For W3 and W4 the performance is the highest for temperatures 0.00, 0.25 and 0.50. The mean macro average F1 score is the highest for temperatures 0.00, 0.25 and 0.50. Based on this experiment, we chose to use a temperature of 0.25 for GPT-4 in the rest of the experiments as it has a marginally higher macro average F1 score than the other temperatures.

	0.00	0.25	0.50	1.00	1.50
W1	<b>0.520</b>	0.507	0.493	0.487	0.487
W2	0.507	<b>0.537</b>	<b>0.537</b>	0.513	0.490
W3	<b>0.737</b>	<b>0.743</b>	<b>0.740</b>	0.683	0.717
W4	<b>0.763</b>	<b>0.763</b>	<b>0.753</b>	0.740	0.710
mean	<b>0.632</b>	<b>0.638</b>	<b>0.631</b>	0.606	0.601

Table 5.10: Mean macro averaged F1 score per temperature per web use case

#### 5.2.3.2 Experiment 3.2: Main Experiment

The results of the third experiment are shown in Table 5.11. The scores show that the GPT-4, TaSeR and S-TaSeR methods excel in different use cases. For the web use cases, GPT-4 ZS performs best with a slightly higher score than the TaSeR methods. For W2, W3 and W4 the TaSeR method performs best, with GPT-4 ZS following closely with a bit lower scores. While scoring lower than TaSeR, GPT-4 ZS does outperform S-TaSeR on these use cases. For the labor market use cases the GPT-4 ZS method outperforms both TaSeR methods, with slightly higher scores. Continuing with the first biomedical use cases, we find that both TaSeR method outperform the GPT-4 ZS method on all use cases except for B3. On B3 both TaSeR methods score a fair amount lower than GPT4 ZS, with the GPT-4 method scoring almost twice as high. If we look at the runtimes, we find that the GPT-4 method is a fair amount slower than the TaSeR methods. The differences in runtime range from 1 minute for W2, up to 72 minutes for L1.

	Random	TaSeR	S-TaSeR	GPT-4 ZS
W1	0.33 (-)	0.44 (2.2)	0.43 (2.2)	<b>0.51</b> (4.8)
W2	0.33 (-)	<b>0.46</b> (0.5)	0.39 (0.5)	0.42 (1.5)
W3	0.33 (-)	<b>0.80</b> (0.6)	0.62 (0.6)	0.77 (2.1)
W4	0.33 (-)	<b>0.83</b> (0.6)	0.47 (0.5)	0.76 (1.8)
L1	0.33 (-)	0.48 (2.6)	0.48 (2.7)	<b>0.54</b> (74.8)
L2	0.33 (-)	0.45 (1.8)	0.40 (1.9)	<b>0.47</b> (38.0)
B1	0.33 (-)	<b>0.90</b> (5.3)	<b>0.90</b> (5.6)	0.68 (68.3)
B2	0.33 (-)	<b>0.58</b> (4.7)	<b>0.58</b> (5.0)	0.42 (37.5)
B3	0.33 (-)	0.34 (27.5)	0.35 (29.4)	<b>0.61</b> (68.3)
B4	0.33 (-)	<b>0.82</b> (5.3)	0.79 (5.5)	0.57 (40.5)
B5	0.33 (-)	<b>0.85</b> (7.3)	0.82 (7.6)	0.57 (68.3)

Table 5.11: Mean macro average F1 score for TaSeR, S-TaSeR and macro average F1 score for random classification and GPT-4 ZS

### 5.2.3.3 Investigating individual runs for experiment 3

In Table 5.12 and 5.13 the confusion matrix and classification report of GPT-4 ZS and the median run of TaSeR are shown. We see that the performance difference between the methods can mainly be attributed to the tendency of TaSeR for over-classifying triples as *exactMatch* as can be seen in the confusion matrix.

		Predicted			Classification Report		
		broadMatch	exactMatch	narrowMatch	Precision	Recall	F1-Score
Actual	broadMatch	830	1028	99	0.70	0.42	0.53
	exactMatch	84	848	57	0.26	0.86	0.40
	narrowMatch	266	1400	863	0.85	0.34	0.49
				Macro avg. F1		0.47	

Table 5.12: Combined confusion matrix and classification report for the median run of the TaSeR runs on use case L1

		Predicted			Classification Report		
		broadMatch	exactMatch	narrowMatch	Precision	Recall	F1-Score
Actual	broadMatch	661	80	1216	0.65	0.34	0.44
	exactMatch	97	387	505	0.72	0.39	0.51
	narrowMatch	256	71	2202	0.56	0.87	0.68
				Macro avg. F1		0.54	

Table 5.13: Combined confusion matrix and classification report for the GPT-4 ZS run on use case L1

In the previous experiments we saw the confusion matrices and classification reports of the TaSeR and S-TaSeR methods for the B3 use case and found that they have a tendency to over-classify triples as *exactMatch*. We display the table for TaSeR again in Table 5.14 for easy comparison. S-TaSeR is left out as the results for the two TaSeR methods is very similar, as can be seen in experiment 2. In Table 5.15 the confusion matrix and classification report of the GPT-4 ZS method on the B3 use case is shown. We find that, in contrast to the TaSeR method, the GPT-4 ZS method has a very stable score across all relations.

		Predicted			Classification Report		
		broadMatch	exactMatch	narrowMatch	Precision	Recall	F1-Score
Actual	broadMatch	235	1272	0	1.00	0.16	0.27
	exactMatch	0	1985	0	0.42	1.00	0.60
	narrowMatch	0	1419	88	1.00	0.06	0.11
				Macro avg. F1		0.33	

Table 5.14: Combined confusion matrix and classification report for one of the 5 TaSeR runs on use case B3

		Predicted			Classification Report		
		broadMatch	exactMatch	narrowMatch	Precision	Recall	F1-Score
Actual	broadMatch	794	45	668	0.77	0.53	0.62
	exactMatch	172	967	846	0.84	0.49	0.62
	narrowMatch	68	144	1295	0.46	0.86	0.60
					Macro avg. F1		0.61

Table 5.15: Combined confusion matrix and classification report for the GPT-4 ZS run on use case B3

#### 5.2.4 Experiment 4: Comparing GPT-4 in a zero-shot and a few-shot setting

The result of experiment 4 are shown in Table 5.16. The scores show that the zero-shot approach generally outperforms the few-shot approach for most use cases. The difference is the largest for the W1, W3, W4 and B2 use cases. For W2 both methods score equally good. The few-shot approach does outperform the zero-shot approach in two use cases: B1 and B4. The runtimes are similar, with the few-shot approach taking slightly longer, with differences ranging from 0.3 minutes for use case W4, to 13.7 for the L1 use case.

	Random	GPT-4 ZS	GPT-4 FS
W1	0.33 (-)	<b>0.51</b> (4.8)	0.39 (5.7)
W2	0.33 (-)	<b>0.42</b> (1.5)	<b>0.43</b> (1.8)
W3	0.33 (-)	<b>0.77</b> (2.1)	0.52 (2.5)
W4	0.33 (-)	<b>0.76</b> (1.8)	0.62 (2.2)
L1	0.33 (-)	<b>0.54</b> (74.8)	0.45 (88.5)
L2	0.33 (-)	<b>0.47</b> (38.0)	0.40 (44.9)
B1	0.33 (-)	0.68 (68.3)	<b>0.73</b> (80.8)
B2	0.33 (-)	<b>0.42</b> (37.5)	0.31 (44.4)
B3	0.33 (-)	<b>0.61</b> (68.3)	0.59 (80.8)
B4	0.33 (-)	<b>0.57</b> (40.5)	0.42 (47.9)
B5	0.33 (-)	<b>0.57</b> (68.3)	0.55 (80.8)

Table 5.16: Macro average F1 score for Random, GPT-4 ZS and GPT-4 FS

##### 5.2.4.1 Investigating individual runs for experiment 4

In tables 5.17 and 5.18 we investigate the difference in performance between the two methods on use case W3, as the difference between them is the largest here. We find that both methods have relatively low scores for the *broadMatch* relation, with GPT-4 FS scoring very low with 0.14. Note however that the number of actual triples with a *broadMatch* relation is very low, as there are only 14 of them.

		Predicted			Classification Report		
		broadMatch	exactMatch	narrowMatch	Precision	Recall	F1-Score
Actual	broadMatch	10	0	4	0.38	0.71	0.50
	exactMatch	3	23	2	1.00	0.82	0.90
	narrowMatch	13	0	99	0.94	0.88	0.91
					Macro avg. F1		0.77

Table 5.17: Combined confusion matrix and classification report for the GPT-4 ZS run on use case W3

		Predicted			Classification Report		
		broadMatch	exactMatch	narrowMatch	Precision	Recall	F1-Score
Actual	broadMatch	6	0	8	0.09	0.43	0.14
	exactMatch	6	20	2	0.95	0.71	0.82
	narrowMatch	57	1	54	0.84	0.48	0.61
					Macro avg. F1		0.52

Table 5.18: Combined confusion matrix and classification report for the GPT-4 FS run on use case W3

In tables 5.19 and 5.20 we have visualised the differences in performance between the two methods for the B2 use case, as the GPT-4 FS method has a lower score than random classification. The tables show that both methods have lower scores for the *broadMatch* and *narrowMatch* relations. However, the FS method has scores of close to zero with 0.08 and 0.09 for *broadMatch* and *narrowMatch* respectively.

		Predicted			Classification Report		
		broadMatch	exactMatch	narrowMatch	Precision	Recall	F1-Score
Actual	broadMatch	14	2	55	0.56	0.20	0.29
	exactMatch	10	1854	738	1.00	0.71	0.83
	narrowMatch	1	3	68	0.08	0.94	0.15
					Macro avg. F1		0.42

Table 5.19: Combined confusion matrix and classification report for the GPT-4 ZS run on use case B2

		Predicted			Classification Report		
		broadMatch	exactMatch	narrowMatch	Precision	Recall	F1-Score
Actual	broadMatch	19	1	51	0.05	0.27	0.08
	exactMatch	353	1561	688	1.00	0.60	0.75
	narrowMatch	30	2	40	0.05	0.56	0.09
					Macro avg. F1		0.31

Table 5.20: Combined confusion matrix and classification report for the GPT-4 FS run on use case B2

## Chapter 6

# Discussion

### 6.1 Discussion of the experimental results

#### 6.1.1 SQ1: How do current mapping refinement methods that incorporate domain knowledge compare to methods that do not?

In the results presented in Section 5.2.1, STROMA exhibited effective performance for the web use cases for which it was developed, yet its performance was worse than random in the remaining use cases. This shows us that the STROMA method is not suitable for applications on other use cases than it was built with, at least not in the labor market and biomedical domains. The TaSeR base method showed worse than random results across all use cases, showing that TaSeR does not perform well without fine-tuning on a specific task. We also saw that TaSeR, when fine-tuned on the ontologies, has the best performance across all use cases. This demonstrates that leveraging domain knowledge is the preferred approach in the current state-of-the-art. The runtime of the TaSeR method were higher than those of the other methods, which is caused by the time the method takes to fine-tune the model. For use cases with a large number of samples in the ontologies, the difference is the largest. We also saw a limitation of the TaSeR method, being that its performance depends on the underlying data and settings during training, specifically for use case B3. Its over-classification of *exactMatch* triples is likely to be caused by overfitting, as the train dataset contains a very large number of triples compared to the other biomedical use cases. As we set the batch size per use case, it might have been too low for the B3 use case. Keeping in mind this limitation, the answer to the subquestion is thus that the state-of-the-art method that uses domain knowledge, TaSeR, outperforms state-of-the-art methods that do not use domain knowledge.

#### 6.1.2 SQ2: Does including additional general knowledge via pre-fine-tuning, as done in the current state of the art of mapping refinement, benefit its performance?

In the results in Section 5.2.2 of the web use cases, we saw that TaSeR performed better than S-TaSeR. This could be caused by the fact that the web use cases have small ontologies, resulting in a small train dataset. This in turn increases the chance that the model does not learn to correctly distinguish between the classes, resulting in a lower accuracy and macro average f1 score. As TaSeR is pre-fine-tuned for relation prediction, it might thus need less data than the S-TaSeR method, which does not include pre-fine-tuning. This need for enough train data is a limitation for the S-TaSeR method. For the labor market use cases all methods perform comparably, with S-TaSeR scoring slightly higher for some. When looking at the variance of the predictions of the two methods, we found that the scores of TaSeR were not only higher, but also more consistent. This shows that TaSeR is

more robust than S-TaSeR and that pre-fine-tuning is therefore beneficial. In the individual use case analysis we found that S-TaSeR suffers from the same problem as TaSeR in the first experiment as it also over-predicts the *exactMatch* relation for the B3 use case. As both models are based on the same DistilBERT model, this is also likely to be caused by overfitting. For the W4 use case we found that S-TaSeR could not differentiate between the *broadMatch* and *narrowMatch* relations as most *broadMatch* triples were classified as *narrowMatch*, while TaSeR was able to correctly classify them. Taking into account that the *narrowMatch* triples are the exact inverses of the *broadMatch* triples for the W4 use case, this shows that S-TaSeR is not able to learn that the order in which the entities are given is important for the relation that it predicts. In combination with the fact that the W4 use case has the smallest train dataset of all use cases, this shows that TaSeR needs less train data in order to be able to differentiate between *broadMatch* and *narrowMatch*. This, in combination that both methods score approximately equivalent on the use cases where there is enough train data, again shows the limitation of S-TaSeR that it needs more train data than TaSeR. With that in mind, we answer the subquestion by concluding that including additional general knowledge via pre-fine-tuning is indeed beneficial for the performance of the method.

### 6.1.3 SQ3: How do state-of-the-art mapping refinement methods compare to generative large language models in a zero-shot setting?

In the temperature tuning experiment in Section 5.2.3.1 we saw that a lower temperature is generally beneficial for the performance of GPT-4 in classification. This is in line with findings in previous work [74]. It is interesting that the temperature closest to a deterministic setting, 0.00, did not result in the best performance, indicating that some randomness in the output is beneficial for its performance.

In the main experiment in Section 5.2.3.2 we found that GPT-4 in a zero-shot setting (ZS) and TaSeR have comparable performance, each excelling in different use cases. For the web use cases we found that GPT-4 ZS outperformed S-TaSeR, but not TaSeR, confirming the conclusion from experiment 2 that S-TaSeR does not perform well when there is only a small amount of train triples available. The fact that TaSeR does outperform GPT-4 ZS on these small-train-dataset use cases also again confirms that pre-fine-tuning lowers the number of required train triples. However, it is good to keep in mind that the web use cases were used during the development of TaSeR. Specifically, the test sets were used for hyper parameter tuning before the model was fine-tuned and published. Therefore, like for STROMA, the results of TaSeR on the web use cases should be interpreted with caution. For the labor market use cases the GPT-4 ZS method outperformed both TaSeR methods. It is interesting to see that the GPT-4 ZS method performs best on the L2 use case, as this includes both ESCO and O\*NET. ESCO and O\*NET were used as a part of the pre-train dataset of GPT-4. CompetentNL has not yet been published and is therefore not a part of the pre-train dataset of GPT-4. The difference in performance between the two labor market use cases could be caused by GPT-4 having internalized both ESCO and O\*NET for the L2 use case, but only ESCO for the L1 use case. This shows that domain knowledge could also have a small influence on the performance on down-stream-tasks, if it is included during pre-training. Comparing the results of the TaSeR methods with the GPT-4 ZS method shows that the TaSeR methods over-classify the *exactMatch* relation. This could be caused by the fact that the *altLabels* from ESCO, used as *exactMatch* triples, are of lower quality, with some *altLabels* containing labels that could also be subclasses of the given entity. For instance, the occupation *Data scientist* has *altLabels* *Data research scientist* and *Data expert*. One could argue that these are subclasses or superclasses, indicating that the *altLabels* are not always synonyms. This shows the dependency of the TaSeR methods on the train

datasets. The results of GPT-4 ZS are consistent across all relations, showing a more robust performance. For the biomedical use cases, the TaSeR methods outperform the GPT-4 ZS method by a large margin. Only for use case B3 this is not the case. The reason likely to cause this, overfitting, is explain in the discussion of experiment 1. The biomedical use cases contain very large ontologies and use highly domain specific terms. The superior results of both TaSeR methods show that training on domain knowledge is still the most effective approach to mapping refinement for such use cases. Concerning the runtime, the TaSeR methods were faster than the GPT-4 ZS method by a large margin. It is however good to keep in mind that only a little pre-processing is required for using the GPT-4 ZS method, while the TaSeR methods require the pre-processing of both ontologies into RDF format.

Based on the results, we conclude that using domain knowledge based methods is still preferred, particularly for use cases where there is a large train dataset available labels are highly domain specific. We add a small side-note to this, as the GPT-4 ZS method has a competitive performance and has the advantages of not requiring large amounts of training data and the process of pre-processing which that data requires. It is however sometimes inconsistent in its formatting, which should be taking into account.

One side-note about the output of GPT-4 is that in the predictions of the GPT-4 method is the presence of oddities in its predictions. For some entity pairs two predicted relations were given separated by a line break. After investigation we found that this was caused by the occurrence of the & character. For instance, the entity pair (*mushrooms & truffles, chocolate truffles*) resulted in the prediction *broadMatch\n narrowMatch*. Other oddities that occurred were cases where the model repeated the entities in its answer like *"relation between concepts: [life coach], and: [professional organiser] is narrowMatch"* and spelling mistakes. These errors however did not have any effect on the results shown in the results section, as we corrected them by hand.

#### 6.1.4 SQ4: Does incorporating domain knowledge into generative large language models through few-shot learning enhance their performance in mapping refinement?

In our last experiment, the results of which can be found in Section 5.2.4, we found that generally using the zero-shot setting outperforms using the few-shot setting. Including examples from the domain ontologies thus has no positive effect on the performance of the GPT-4 method. The runtime is also increased by a bit when using the few-shot setting, making it an even less attractive option. One positive aspect of the few-shot setting was that the previously described oddities in the predictions of the GPT-4 in zero-shot setting occurred far less. While not perfect, the oddities were significantly less, with only a couple of irregularities over all use cases. The answer to the last subquestion is that, according to our results, including domain knowledge in generative large language models via few-shot learning is not beneficial to the performance of these models on mapping refinement.

These findings are not in line with literature. Many sources indicate that using a few-shot setting instead of a zero-shot setting is beneficial for the results [76, 66, 61]. A reason why our experiments indicate the opposite might be the way the few-shot prompts are formatted. Literature suggests that the exact formatting of the prompt has a large effect on the performance [46, 71, 59]. We did experiment with various prompt formats and chose the best performing one, but more experimentation could result in better performance.

## 6.2 Discussion of the application of mapping refinement

Within the project this thesis is part of, a graphical user interface (GUI) tool has been developed. It is designed to facilitate the creation of refined mappings between two ontologies. This GUI tool is flexible, supporting the integration of any two ontologies. Its primary

function is to aid users in accurately mapping entities between the CompetentNL ontology and various labor market ontologies. The tool’s anticipated broader application includes supporting SBB<sup>1</sup>, a partner company of TNO, by enhancing their efficiency in generating refined mappings.

Following our quantitative experiments, we decided to incorporate the GPT-4 zero-shot method into the GUI as it was the best performing method for the labor market use cases. This method is used in combination with a matcher method to guide users in the mapping creation process, offering suggested target entities for each source entity selected. A visual representation of the GUI is provided in Figure 6.1. To protect intellectual property, the CompetentNL ontology is intentionally blurred in this image due to its pending publication. In the GUI, users select an entity from the source ontology displayed on the left, which triggers the display of a list of suggested target entities on the right. These suggestions include the top four predictions, ranked by the confidence scores determined by the matcher. Next to each suggested target entity’s name, five relational options are presented: *broadMatch*, *exactMatch*, *narrowMatch*, *other* and *no match*. The *other* option accommodates potential matches that don’t fit into predefined categories and the *no match* option identifies unrelated entities, valuable for generating true negatives for future training improvements.

The GUI suggests which relation to select for each mapping, with the recommended choice visually indicated by a blue circle surrounding the relevant relation button. This intuitive interface simplifies the mapping process, ensuring it is clear and user-friendly. Since the creation of the GUI, multiple experiments have been conducted in collaboration with SBB. The GUI has been received positively, with users complimenting the completeness and ease of use of the tool.

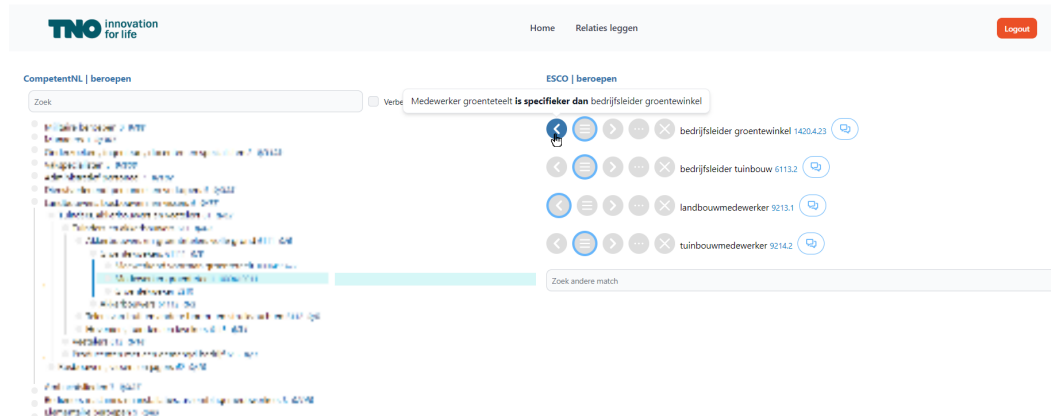


Figure 6.1: A snapshot of the GUI where the GPT-4 method will be used in

<sup>1</sup><https://www.s-bb.nl>

# Chapter 7

## Conclusion

### 7.1 Summary of the results

In this thesis we investigated the impact of incorporating domain knowledge into mapping refinement methods and compared these methods to generative large language models in both zero-shot and few-shot settings. Our findings demonstrate that domain knowledge plays a crucial role in enhancing the performance of mapping refinement methods. The TaSeR method, when fine-tuned with domain-specific ontologies, outperformed other state-of-the-art methods that do not leverage domain knowledge, particularly in scenarios with a large number of samples. This underscores the importance of domain knowledge in achieving high accuracy and robustness in semantic mapping tasks. Moreover, we identified limitations related to overfitting due to a too small batch size and dependency on the quality and size of the training data, highlighting areas for future improvements.

The comparison between state-of-the-art mapping refinement methods and generative large language models revealed that, despite the competitive performance of GPT-4 in a zero-shot setting, incorporating domain knowledge through fine-tuning a BERT model remains the most effective approach, especially for complex use cases with domain-specific terminologies. However, the GPT-4 zero-shot method's flexibility and reduced need for extensive training data present it as a valuable alternative in certain contexts.

Interestingly, our experiments showed that few-shot learning does not necessarily enhance the performance of generative large language models in mapping refinement tasks. This contradicts existing literature and suggests that the effectiveness of few-shot learning may be highly dependent on the specific task and the manner in which domain knowledge is integrated into the training process.

### 7.2 Summary of the main contributions

#### 7.2.1 Implications for Researchers

The findings from this thesis offer several implications for researchers in the fields of semantic mapping, ontology alignment and artificial intelligence. Firstly, our research confirms the importance of incorporating domain knowledge into mapping refinement methods, highlighting its role in achieving good performance. This emphasizes the need for further investigation into efficient and scalable ways to integrate domain-specific information into semantic mapping algorithms. Secondly, it shows the trade-off between ease of use of generative large language models with slightly lower scores versus higher performance but longer pre-processing for domain knowledge methods. This work also shows that a rule-based approach like STROMA does not perform well on datasets other than it was created with, which may be an indication of the limited generalizability that rule-based approaches face.

### 7.2.2 Implications for Practitioners

For practitioners who are looking to use mapping refinement, this thesis provides valuable insights into the selection and implementation of various methods. The demonstrated superiority of the TaSeR method when fine-tuned with domain knowledge suggests that practitioners should prioritize methods that are able to include such domain knowledge. Moreover, the practical deployment of our findings in a GUI tool underscores the importance of user-friendly interfaces in the adoption of semantic mapping technologies. It also serves as an example of how mapping refinement should be used, as techniques that give suggestions instead of directly deciding upon the result, as its performance is not good enough to be used in a automatic configuration. In Figure 7.1 we have displayed a decision tree for deciding which mapping refinement method to use.

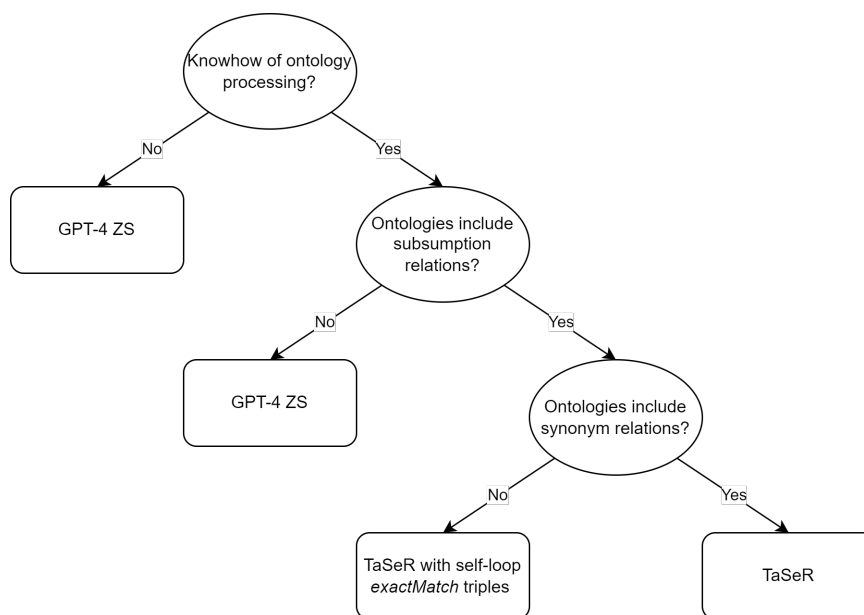


Figure 7.1: A decision tree for deciding which mapping refinement method to use

## 7.3 Limitations

In this thesis the research was performed as correct as possible. However, there are some limitations that we need to address. One limitations exist with regards to the data. The introduction of inverse *broadMatch* and *narrowMatch* triples in the W4 and biomedical use cases might have brought some bias into the datasets. It might have had effects that were not accounted for in our conclusions.

A limitation with regards to the experimental setup is that that the use of small datasets for temperature tuning with GPT-4 might not have fully captured the complexity of typical use cases. Larger datasets could lead to different insights regarding optimal temperature settings. A second limitation with regards to the experimental setup is that the TaSeR and S-TaSeR methods were run only five times to account for randomness in the DistilBERT model. Increasing the number of runs could provide a more solid foundation for our findings. Another limitation with regards to the experimental setup is the absence of Dhouib et al.'s method [104] from our analysis, due to unavailable code and no response after reaching out to the author. If we could have included the method, the comparative analysis would be more exhaustive. A last limitation with regards to the experimental setup is related to the financial constraints associated with the GPT-4 API. Due to these restrictions we conducted

a singular run for these models, which results in the results being subject to the randomness inherent in generative large language models. These limitations should be taken into account when interpreting our results.

## 7.4 Future work

Future research could focus several on several paths to enhance mapping refinement methods. The first path of future work that comes to mind is the exploration of other generative large language models. Conducting comparative analyses of GPT-4 against these other models like Mistral and LLama2 could lead to interesting results.

Another path would be the exploration of a method that combines an encoder based model like BERT with a generative large language model. An example of this is the aforementioned Olala method[44], which combines a BERT model with a Llama2 model, resulting in a good performing method.

Finally, regarding the architecture of the TaSeR method, adjustments could be explored. One approach could be to substitute BERT with a newer embedding model such as Mistral. Another approach would be to experiment with various architectures for the classification layer, as the current setup uses a relatively simple classifier on top of the BERT embeddings.

And with that in mind, we see that there will forever be new paths for future research, a truth that holds not only for ontology alignment but for the vast and limitless landscape of science as a whole.

# Bibliography

- [1] “ISTAT - NOMENCLATURA E CLASSIFICAZIONE DELLE UNITÀ PROFESSIONALI.” [Online]. Available: <https://professioni.istat.it/sistemainformativoprofessionioni/cp2011/?db=2021>
- [2] “ESCO,” Jun. 2023. [Online]. Available: <https://esco.ec.europa.eu/nl>
- [3] J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat, R. Avila, I. Babuschkin, S. Balaji, V. Balcom, P. Baltescu, H. Bao, M. Bavarian, J. Belgum, I. Bello, J. Berdine, G. Bernadett-Shapiro, C. Berner, L. Bogdonoff, O. Boiko, M. Boyd, A.-L. Brakman, G. Brockman, T. Brooks, M. Brundage, K. Button, T. Cai, R. Campbell, A. Cann, B. Carey, C. Carlson, R. Carmichael, B. Chan, C. Chang, F. Chantzis, D. Chen, S. Chen, R. Chen, J. Chen, M. Chen, B. Chess, C. Cho, C. Chu, H. W. Chung, D. Cummings, J. Currier, Y. Dai, C. Decareaux, T. Degry, N. Deutsch, D. Deville, A. Dhar, D. Dohan, S. Dowling, S. Dunning, A. Ecoffet, A. Eleti, T. Eloundou, D. Farhi, L. Fedus, N. Felix, S. P. Fishman, J. Forte, I. Fulford, L. Gao, E. Georges, C. Gibson, V. Goel, T. Gogineni, G. Goh, R. Gontijo-Lopes, J. Gordon, M. Grafstein, S. Gray, R. Greene, J. Gross, S. S. Gu, Y. Guo, C. Hallacy, J. Han, J. Harris, Y. He, M. Heaton, J. Heidecke, C. Hesse, A. Hickey, W. Hickey, P. Hoeschele, B. Houghton, K. Hsu, S. Hu, X. Hu, J. Huizinga, S. Jain, S. Jain, J. Jang, A. Jiang, R. Jiang, H. Jin, D. Jin, S. Jomoto, B. Jonn, H. Jun, T. Kaftan, Kaiser, A. Kamali, I. Kanitscheider, N. S. Keskar, T. Khan, L. Kilpatrick, J. W. Kim, C. Kim, Y. Kim, H. Kirchner, J. Kiros, M. Knight, D. Kokotajlo, Kondraciuk, A. Kondrich, A. Konstantinidis, K. Kosic, G. Krueger, V. Kuo, M. Lampe, I. Lan, T. Lee, J. Leike, J. Leung, D. Levy, C. M. Li, R. Lim, M. Lin, S. Lin, M. Litwin, T. Lopez, R. Lowe, P. Lue, A. Makanju, K. Malfacini, S. Manning, T. Markov, Y. Markovski, B. Martin, K. Mayer, A. Mayne, B. McGrew, S. M. McKinney, C. McLeavey, P. McMillan, J. McNeil, D. Medina, A. Mehta, J. Menick, L. Metz, A. Mishchenko, P. Mishkin, V. Monaco, E. Morikawa, D. Mossing, T. Mu, M. Murati, O. Murk, D. Mély, A. Nair, R. Nakano, R. Nayak, A. Neelakantan, R. Ngo, H. Noh, L. Ouyang, C. O’Keefe, J. Pachocki, A. Paino, J. Palermo, A. Pantuliano, G. Parascandolo, J. Parish, E. Parparita, A. Passos, M. Pavlov, A. Peng, A. Perelman, F. d. A. B. Peres, M. Petrov, H. P. d. O. Pinto, Michael, Pokorny, M. Pokrass, V. Pong, T. Powell, A. Power, B. Power, E. Proehl, R. Puri, A. Radford, J. Rae, A. Ramesh, C. Raymond, F. Real, K. Rimbach, C. Ross, B. Rotsted, H. Roussez, N. Ryder, M. Saltarelli, T. Sanders, S. Santurkar, G. Sastry, H. Schmidt, D. Schnurr, J. Schulman, D. Selsam, K. Sheppard, T. Sherbakov, J. Shieh, S. Shoker, P. Shyam, S. Sidor, E. Sigler, M. Simens, J. Sitkin, K. Slama, I. Sohl, B. Sokolowsky, Y. Song, N. Staudacher, F. P. Such, N. Summers, I. Sutskever, J. Tang, N. Tezak, M. Thompson, P. Tillet, A. Tootoonchian, E. Tseng, P. Tuggle, N. Turley, J. Tworek, J. F. C. Uribe, A. Vallone, A. Vijayvergiya, C. Voss, C. Wainwright, J. J. Wang, A. Wang, B. Wang, J. Ward, J. Wei, C. J. Weinmann, A. Welihinda, P. Welinder, J. Weng, L. Weng, M. Wiethoff, D. Willner, C. Winter, S. Wolrich, H. Wong, L. Workman, S. Wu, J. Wu, M. Wu, K. Xiao, T. Xu, S. Yoo, K. Yu,

- Q. Yuan, W. Zaremba, R. Zellers, C. Zhang, M. Zhang, S. Zhao, T. Zheng, J. Zhuang, W. Zhuk, and B. Zoph, “GPT-4 Technical Report,” Tech. Rep., 2023, \_eprint: 2303.08774. [Online]. Available: <https://arxiv.org/abs/2303.08774>
- [4] A. Aghajanyan, A. Gupta, A. Shrivastava, X. Chen, L. Zettlemoyer, and S. Gupta, “Muppet: Massive Multi-task Representations with Pre-Finetuning,” in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, M.-F. Moens, X. Huang, L. Specia, and S. W.-t. Yih, Eds. Online and Punta Cana, Dominican Republic: Association for Computational Linguistics, Nov. 2021, pp. 5799–5811. [Online]. Available: <https://aclanthology.org/2021.emnlp-main.468>
- [5] M. Amir, M. Baruah, M. Eslamialishah, S. Ehsani, A. Bahramali, S. Naddaf-Sh, and S. Zarandioon, “Truveta Mapper: A Zero-shot Ontology Alignment Framework,” 2023, \_eprint: 2301.09767. [Online]. Available: <https://arxiv.org/abs/2301.09767>
- [6] P. Arnold, “Semantic Enrichment of Ontology Mappings.” Dec. 2015. [Online]. Available: <https://ul.qucosa.de/api/qucosa%3A14126/attachment/ATT-0/>
- [7] P. Arnold and E. Rahm, “Enriching ontology mappings with semantic relations,” *Data & Knowledge Engineering*, vol. 93, pp. 1–18, Sep. 2014. [Online]. Available: <https://linkinghub.elsevier.com/retrieve/pii/S0169023X14000603>
- [8] C. Batini, M. Lenzerini, and S. B. Navathe, “A comparative analysis of methodologies for database schema integration,” *ACM Computing Surveys*, vol. 18, no. 4, pp. 323–364, Dec. 1986. [Online]. Available: <https://dl.acm.org/doi/10.1145/27633.27634>
- [9] S. Bechhofer, F. v. Harmelen, Frank van, J. Hendler, I. Horrocks, D. L. McGuinness, P. F. Patel-Schneider, and L. Anrea Stein, “OWL Web Ontology Language Reference,” Feb. 2004. [Online]. Available: <https://www.w3.org/TR/owl-ref/>
- [10] R. Blanco, B. B. Cambazoglu, P. Mika, and N. Torzecz, “Entity Recommendations in Web Search,” in *The Semantic Web – ISWC 2013*, D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, H. Alani, L. Kagal, A. Fokoue, P. Groth, C. Biemann, J. X. Parreira, L. Aroyo, N. Noy, C. Welty, and K. Janowicz, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, vol. 8219, pp. 33–48, series Title: Lecture Notes in Computer Science. [Online]. Available: [http://link.springer.com/10.1007/978-3-642-41338-4\\_3](http://link.springer.com/10.1007/978-3-642-41338-4_3)
- [11] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, “Enriching Word Vectors with Subword Information,” *Transactions of the Association for Computational Linguistics*, vol. 5, pp. 135–146, Jun. 2017, \_eprint: [https://direct.mit.edu/tacl/article-pdf/doi/10.1162/tacl\\_a\\_00051/1567442/tacl\\_a\\_00051.pdf](https://direct.mit.edu/tacl/article-pdf/doi/10.1162/tacl_a_00051/1567442/tacl_a_00051.pdf). [Online]. Available: [https://doi.org/10.1162/tacl\\_a\\_00051](https://doi.org/10.1162/tacl_a_00051)
- [12] P. A. Bonatti, S. Decker, A. Polleres, and V. Presutti, “Knowledge Graphs: New Directions for Knowledge Representation on the Semantic Web (Dagstuhl Seminar 18371),” p. 83 pages, 2019, artwork Size: 83 pages Medium: application/pdf Publisher: Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik GmbH, Wadern/Saarbruecken, Germany Version Number: 1.0. [Online]. Available: <http://drops.dagstuhl.de/opus/volltexte/2019/10328/>
- [13] P. Bouquet, B. Magnini, L. Serafini, and S. Zanolini, “A SAT-Based Algorithm for Context Matching,” in *Modeling and Using Context*, P. Blackburn, C. Ghidini, R. M. Turner, and F. Giunchiglia, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, vol. 2680, pp. 66–79, series Title: Lecture Notes in Computer Science. [Online]. Available: [http://link.springer.com/10.1007/3-540-44958-2\\_6](http://link.springer.com/10.1007/3-540-44958-2_6)

- [14] P. Bouquet, L. Serafini, and S. Zanobini, "Semantic Coordination: A New Approach and an Application," in *The Semantic Web - ISWC 2003*, G. Goos, J. Hartmanis, J. Van Leeuwen, D. Fensel, K. Sycara, and J. Mylopoulos, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, vol. 2870, pp. 130–145, series Title: Lecture Notes in Computer Science. [Online]. Available: [http://link.springer.com/10.1007/978-3-540-39718-2\\_9](http://link.springer.com/10.1007/978-3-540-39718-2_9)
- [15] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, "Language Models are Few-Shot Learners," in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, Eds., vol. 33. Curran Associates, Inc., 2020, pp. 1877–1901. [Online]. Available: [https://proceedings.neurips.cc/paper\\_files/paper/2020/file/1457c0d6bfc4967418bfb8ac142f64a-Paper.pdf](https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfc4967418bfb8ac142f64a-Paper.pdf)
- [16] G. Brunello and P. Wruuck, "Skill Shortages and Skill Mismatch in Europe: A Review of the Literature," Rochester, NY, May 2019. [Online]. Available: <https://papers.ssrn.com/abstract=3390340>
- [17] J. Chen, Y. He, Y. Geng, E. Jiménez-Ruiz, H. Dong, and I. Horrocks, "Contextual semantic embeddings for ontology subsumption prediction," *World Wide Web*, vol. 26, no. 5, pp. 2569–2591, Sep. 2023. [Online]. Available: <https://doi.org/10.1007/s11280-023-01169-9>
- [18] Q. Chen, J. Du, Y. Hu, V. K. Keloth, X. Peng, K. Raja, Q. Xie, A. Gilson, M. Singer, R. A. Adelman, R. Zhang, and Z. Lu, "A systematic evaluation of large language models for biomedical natural language processing: benchmarks, baselines, and recommendations," 2023. [Online]. Available: <https://arxiv.org/ftp/arxiv/papers/2305/2305.16326.pdf>
- [19] Z. Chen, Y. Wang, B. Zhao, J. Cheng, X. Zhao, and Z. Duan, "Knowledge Graph Completion: A Review," *IEEE Access*, vol. 8, pp. 192 435–192 456, 2020. [Online]. Available: <https://ieeexplore.ieee.org/document/9220143>
- [20] P. F. Christiano, J. Leike, T. Brown, M. Martic, S. Legg, and D. Amodei, "Deep Reinforcement Learning from Human Preferences," in *Advances in Neural Information Processing Systems*, vol. 30. Curran Associates, Inc., 2017. [Online]. Available: [https://papers.nips.cc/paper\\_files/paper/2017/hash/d5e2c0adad503c91f91df240d0cd4e49-Abstract.html](https://papers.nips.cc/paper_files/paper/2017/hash/d5e2c0adad503c91f91df240d0cd4e49-Abstract.html)
- [21] I. F. Cruz, F. P. Antonelli, and C. Stroe, "AgreementMaker: efficient matching for large real-world schemas and ontologies," *Proceedings of the VLDB Endowment*, vol. 2, no. 2, pp. 1586–1589, Aug. 2009. [Online]. Available: <https://dl.acm.org/doi/10.14778/1687553.1687598>
- [22] J. David, F. Guillet, and H. Briand, "Matching directories and OWL ontologies with AROMA," in *Proceedings of the 15th ACM international conference on Information and knowledge management - CIKM '06*. Arlington, Virginia, USA: ACM Press, 2006, p. 830. [Online]. Available: <http://portal.acm.org/citation.cfm?doid=1183614.1183752>
- [23] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman, "Indexing by latent semantic analysis," *Journal of the American Society for Information Science*, vol. 41, no. 6, pp. 391–407, Sep. 1990. [Online]. Available: [https://onlinelibrary.wiley.com/doi/10.1002/\(SICI\)1097-4571\(199009\)41:6<391::AID-AS11>3.0.CO;2-9](https://onlinelibrary.wiley.com/doi/10.1002/(SICI)1097-4571(199009)41:6<391::AID-AS11>3.0.CO;2-9)

- 
- [24] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding,” in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, J. Burstein, C. Doran, and T. Solorio, Eds. Minneapolis, Minnesota: Association for Computational Linguistics, Jun. 2019, pp. 4171–4186. [Online]. Available: <https://aclanthology.org/N19-1423>
- [25] A. Doan and A. Y. Halevy, “Semantic Integration Research in the Database Community: A Brief Survey,” *AI Magazine*, vol. 26, no. 1, p. 83, Mar. 2005. [Online]. Available: <https://ojs.aaai.org/aimagazine/index.php/aimagazine/article/view/1801>
- [26] L. Ehrlinger and W. Wöß, “Towards a Definition of Knowledge Graphs,” Sep. 2016. [Online]. Available: <https://ceur-ws.org/Vol-1695/paper4.pdf?ref=https://githubhelp.com>
- [27] J. Euzenat, D. Loup, M. Touzani, and P. Valtchev, “Ontology alignment with OLA,” in *Proc. 3rd ISWC2004 workshop on Evaluation of Ontology-based tools (EON)*, ser. Proc. 3rd ISWC2004 workshop on Evaluation of Ontology-based tools (EON). Hiroshima, Japan: No commercial editor., Nov. 2004, pp. 59–68. [Online]. Available: <https://inria.hal.science/hal-00918128>
- [28] J. Euzenat and P. Shvaiko, *Ontology Matching*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013. [Online]. Available: <https://link.springer.com/10.1007/978-3-642-38721-0>
- [29] G. Fatouros, J. Soldatos, K. Kouroumalis, G. Makridis, and D. Kyriazis, “Transforming sentiment analysis in the financial domain with ChatGPT,” *Machine Learning with Applications*, vol. 14, p. 100508, Dec. 2023. [Online]. Available: <https://linkinghub.elsevier.com/retrieve/pii/S2666827023000610>
- [30] A. Giabelli, L. Malandri, F. Mercorio, and M. Mezzanzanica, “WETA: Automatic taxonomy alignment via word embeddings,” *Computers in Industry*, vol. 138, p. 103626, 2022. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0166361522000215>
- [31] J.-B. Gidrol, “Text classification with LLMs,” Tech. Rep., 2023. [Online]. Available: [https://jbgidrol.com/paper/Text\\_classification\\_with\\_LLMs.pdf](https://jbgidrol.com/paper/Text_classification_with_LLMs.pdf)
- [32] F. Giunchiglia, P. Shvaiko, and M. Yatskevich, “S-Match: an Algorithm and an Implementation of Semantic Matching,” in *The Semantic Web: Research and Applications*, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, O. Nierstrasz, C. Pandu Rangan, B. Steffen, D. Terzopoulos, D. Tygar, M. Y. Vardi, C. J. Bussler, J. Davies, D. Fensel, and R. Studer, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, vol. 3053, pp. 61–75, series Title: Lecture Notes in Computer Science. [Online]. Available: [http://link.springer.com/10.1007/978-3-540-25956-5\\_5](http://link.springer.com/10.1007/978-3-540-25956-5_5)
- [33] —, “Semantic Schema Matching,” in *On the Move to Meaningful Internet Systems 2005: CoopIS, DOA, and ODBASE*, R. Meersman and Z. Tari, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 347–365. [Online]. Available: [https://doi.org/10.1007/11575771\\_23](https://doi.org/10.1007/11575771_23)
- [34] F. Green and G. Henseke, “Europe’s evolving graduate labour markets: supply, demand, underemployment and pay,” *Journal for Labour Market Research*, vol. 55, no. 1, p. 2, Dec. 2021. [Online]. Available: <https://labourmarketresearch.springeropen.com/articles/10.1186/s12651-021-00288-y>

- [35] D. Gromann and T. Declerck, “Comparing Pretrained Multilingual Word Embeddings on an Ontology Alignment Task,” in *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, N. Calzolari, K. Choukri, C. Cieri, T. Declerck, S. Goggi, K. Hasida, H. Isahara, B. Maegaard, J. Mariani, H. Mazo, A. Moreno, J. Odijk, S. Piperidis, and T. Tokunaga, Eds. Miyazaki, Japan: European Language Resources Association (ELRA), May 2018. [Online]. Available: <https://aclanthology.org/L18-1034>
- [36] G. A. Guerra, H. Hofmann, S. Sobhani, G. Hofmann, D. Gomez, D. Soroudi, B. S. Hopkins, J. Dallas, D. J. Pangal, S. Cheok, V. N. Nguyen, W. J. Mack, and G. Zada, “GPT-4 Artificial Intelligence Model Outperforms ChatGPT, Medical Students, and Neurosurgery Residents on Neurosurgery Written Board-Like Questions,” *World Neurosurgery*, vol. 179, pp. e160–e165, Nov. 2023. [Online]. Available: <https://linkinghub.elsevier.com/retrieve/pii/S1878875023011440>
- [37] S. Guru Rao, L. Oosterheert, R. Bakker, S. Wang, N. Strisciuglio, and M. Theune, “Ontology Matching using Background Knowledge and Semantic Similarity,” Ph.D. dissertation, 2022. [Online]. Available: [https://essay.utwente.nl/90521/1/Guru\\_Rao\\_MSc\\_EEMCS.pdf](https://essay.utwente.nl/90521/1/Guru_Rao_MSc_EEMCS.pdf)
- [38] S. Gururangan, A. Marasović, S. Swayamdipta, K. Lo, I. Beltagy, D. Downey, and N. A. Smith, “Don’t Stop Pretraining: Adapt Language Models to Domains and Tasks,” in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, D. Jurafsky, J. Chai, N. Schluter, and J. Tetreault, Eds. Online: Association for Computational Linguistics, Jul. 2020, pp. 8342–8360. [Online]. Available: <https://aclanthology.org/2020.acl-main.740>
- [39] A. Gómez-Pérez, “Ontology Evaluation,” in *Handbook on Ontologies*, ser. International Handbooks on Information Systems, S. Staab and R. Studer, Eds. Berlin, Heidelberg: Springer, 2004, pp. 251–273. [Online]. Available: [https://doi.org/10.1007/978-3-540-24750-0\\_13](https://doi.org/10.1007/978-3-540-24750-0_13)
- [40] F. Hamdi, B. Safar, N. B. Niraula, and C. Reynaud, “TaxoMap alignment and refinement modules: results for OAEI 2010,” in *Proceedings of the 5th International Conference on Ontology Matching - Volume 689*, ser. OM’10. Aachen, DEU: CEUR-WS.org, 2010, pp. 212–219, event-place: Shanghai, China. [Online]. Available: [https://ceur-ws.org/Vol-689/oaei10\\_paper13.pdf](https://ceur-ws.org/Vol-689/oaei10_paper13.pdf)
- [41] Y. He, J. Chen, D. Antonyrajah, and I. Horrocks, “Biomedical ontology alignment with BERT,” in *The 16th International Workshop on Ontology Matching (OM@ISWC-2021)*, vol. 3063, 2021, pp. 1–12. [Online]. Available: <https://ora.ox.ac.uk/objects/uuid:55395994-d156-4744-9dd6-b8b499e1db29>
- [42] —, “BERTMap: A BERT-Based Ontology Alignment System,” *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, no. 5, pp. 5684–5691, Jun. 2022. [Online]. Available: <https://ojs.aaai.org/index.php/AAAI/article/view/20510>
- [43] Y. He, J. Chen, H. Dong, and I. Horrocks, “Exploring Large Language Models for Ontology Alignment,” 2023. [Online]. Available: <http://arxiv.org/abs/2309.07172>
- [44] S. Hertling and H. Paulheim, “OLaLa: Ontology Matching with Large Language Models,” in *Proceedings of the 12th Knowledge Capture Conference 2023*. Pensacola FL USA: ACM, Dec. 2023, pp. 131–139. [Online]. Available: <https://dl.acm.org/doi/10.1145/3587259.3627571>

- 
- [45] —, “Transformer Based Semantic Relation Typing for Knowledge Graph Integration,” in *The Semantic Web*, C. Pesquita, E. Jimenez-Ruiz, J. McCusker, D. Faria, M. Dragoni, A. Dimou, R. Troncy, and S. Hertling, Eds. Cham: Springer Nature Switzerland, 2023, pp. 105–121. [Online]. Available: [https://link.springer.com/chapter/10.1007/978-3-031-33455-9\\_7](https://link.springer.com/chapter/10.1007/978-3-031-33455-9_7)
- [46] T. Heston and C. Khun, “Prompt Engineering in Medical Education,” *International Medical Education*, vol. 2, no. 3, pp. 198–205, Aug. 2023. [Online]. Available: <https://www.mdpi.com/2813-141X/2/3/19>
- [47] G. Hinton, “Learning distributed representations of concepts,” *Proceedings of 8th meeting of the Cognitive Science Society*, vol. 1, 1986. [Online]. Available: <http://www.cs.toronto.edu/~hinton/absps/families.pdf>
- [48] V. Hofmann, J. Pierrehumbert, and H. Schütze, “Dynamic Contextualized Word Embeddings,” in *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, C. Zong, F. Xia, W. Li, and R. Navigli, Eds. Online: Association for Computational Linguistics, Aug. 2021, pp. 6970–6984. [Online]. Available: <https://aclanthology.org/2021.acl-long.542>
- [49] W. Hu and Y. Qu, “Falcon-AO: A practical ontology matching system,” *Journal of Web Semantics*, vol. 6, no. 3, pp. 237–239, Sep. 2008. [Online]. Available: <https://linkinghub.elsevier.com/retrieve/pii/S1570826808000267>
- [50] B. L. Humphreys and D. A. Lindberg, “The UMLS project: making the conceptual connection between users and the information they need.” *Bulletin of the Medical Library Association*, vol. 81, no. 2, pp. 170–177, Apr. 1993. [Online]. Available: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC225759/>
- [51] P. Jain, P. Hitzler, A. P. Sheth, K. Verma, and P. Z. Yeh, “Ontology Alignment for Linked Open Data,” in *The Semantic Web – ISWC 2010*, P. F. Patel-Schneider, Y. Pan, P. Hitzler, P. Mika, L. Zhang, J. Z. Pan, I. Horrocks, and B. Glimm, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, vol. 6496, pp. 402–417, series Title: Lecture Notes in Computer Science. [Online]. Available: [https://link.springer.com/10.1007/978-3-642-17746-0\\_26](https://link.springer.com/10.1007/978-3-642-17746-0_26)
- [52] P. Jain, P. Z. Yeh, K. Verma, R. G. Vasquez, M. Damova, P. Hitzler, and A. P. Sheth, “Contextual Ontology Alignment of LOD with an Upper Ontology: A Case Study with Proton,” in *The Semantic Web: Research and Applications*, G. Antoniou, M. Grobelnik, E. Simperl, B. Parsia, D. Plexousakis, P. De Leenheer, and J. Pan, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, vol. 6643, pp. 80–92, series Title: Lecture Notes in Computer Science. [Online]. Available: [http://link.springer.com/10.1007/978-3-642-21034-1\\_6](http://link.springer.com/10.1007/978-3-642-21034-1_6)
- [53] Y. R. Jean-Mary, E. P. Shironoshita, and M. R. Kabuka, “Ontology matching with semantic verification,” *Journal of Web Semantics*, vol. 7, no. 3, pp. 235–251, Sep. 2009. [Online]. Available: <https://linkinghub.elsevier.com/retrieve/pii/S1570826809000146>
- [54] Y. Kalfoglou and M. Schorlemmer, “Ontology mapping: the state of the art,” *The Knowledge Engineering Review*, vol. 18, no. 1, pp. 1–31, Jan. 2003. [Online]. Available: [https://www.cambridge.org/core/product/identifier/S0269888903000651/type/journal\\_article](https://www.cambridge.org/core/product/identifier/S0269888903000651/type/journal_article)
- [55] K. S. Kalyan, “A survey of GPT-3 family large language models including ChatGPT and GPT-4,” *Natural Language Processing Journal*, vol. 6, p. 100048, Mar. 2024. [Online]. Available: <https://linkinghub.elsevier.com/retrieve/pii/S2949719123000456>

- 
- [56] M. Kejriwal, *Domain-Specific Knowledge Graph Construction*, ser. SpringerBriefs in Computer Science. Cham: Springer International Publishing, 2019. [Online]. Available: <http://link.springer.com/10.1007/978-3-030-12375-8>
- [57] H. Khan, M. Saqib, H. A. Khattak, S. I. Ali, and S. Lee, “Ontology Alignment for Accurate Ontology Matching: A Survey,” in *Digital Health Transformation, Smart Ageing, and Managing Disability*, K. Jongbae, M. Mokhtari, H. Aloulou, B. Abdulrazak, and L. Seungbok, Eds. Cham: Springer Nature Switzerland, 2023, vol. 14237, pp. 338–349, series Title: Lecture Notes in Computer Science. [Online]. Available: [https://link.springer.com/10.1007/978-3-031-43950-6\\_31](https://link.springer.com/10.1007/978-3-031-43950-6_31)
- [58] S. Khan and M. Safyan, “Semantic matching in hierarchical ontologies,” *Journal of King Saud University - Computer and Information Sciences*, vol. 26, no. 3, pp. 247–257, Sep. 2014. [Online]. Available: <https://linkinghub.elsevier.com/retrieve/pii/S1319157814000111>
- [59] A. Khatun and D. Brown, “Reliability Check: An Analysis of GPT-3’s Response to Sensitive Topics and Prompt Wording,” in *Proceedings of the 3rd Workshop on Trustworthy Natural Language Processing (TrustNLP 2023)*, A. Ovalle, K.-W. Chang, N. Mehrabi, Y. Pruksachatkun, A. Galystan, J. Dhamala, A. Verma, T. Cao, A. Kumar, and R. Gupta, Eds. Toronto, Canada: Association for Computational Linguistics, Jul. 2023, pp. 73–95. [Online]. Available: <https://aclanthology.org/2023.trustnlp-1.8/>
- [60] G. Klyne and J. J. Carroll, “Resource Description Framework (RDF): Concepts and Abstract Syntax,” 2004, publisher: W3C Published: W3C Recommendation. [Online]. Available: <http://www.w3.org/TR/2004/REC-rdf-concepts-20040210/>
- [61] J. Kocoń, I. Cichecki, O. Kaszyca, M. Kochanek, D. Szydło, J. Baran, J. Bielaniewicz, M. Gruza, A. Janz, K. Kanclerz, A. Kocoń, B. Koptyra, W. Mieszczewicz-Kowszewicz, P. Miłkowski, M. Oleksy, M. Piasecki, Radliński, K. Wojtasik, S. Woźniak, and P. Kazienko, “ChatGPT: Jack of all trades, master of none,” *Information Fusion*, vol. 99, p. 101861, Nov. 2023. [Online]. Available: <https://linkinghub.elsevier.com/retrieve/pii/S156625352300177X>
- [62] P. Kolyvakis, A. Kalousis, and D. Kiritsis, “DeepAlignment: Unsupervised Ontology Matching with Refined Word Vectors,” in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. New Orleans, Louisiana: Association for Computational Linguistics, 2018, pp. 787–798. [Online]. Available: <http://aclweb.org/anthology/N18-1072>
- [63] P. Lambrix and H. Tan, “SAMBO—A system for aligning and merging biomedical ontologies,” *Journal of Web Semantics*, vol. 4, no. 3, pp. 196–206, Sep. 2006. [Online]. Available: <https://linkinghub.elsevier.com/retrieve/pii/S1570826806000151>
- [64] J. Li, J. Tang, Y. Li, and Q. Luo, “RiMOM: A Dynamic Multistrategy Ontology Alignment Framework,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 21, no. 8, pp. 1218–1232, Aug. 2009, conference Name: IEEE Transactions on Knowledge and Data Engineering. [Online]. Available: <https://ieeexplore.ieee.org/document/4633358>
- [65] Y. Lin, Z. Liu, M. Sun, Y. Liu, and X. Zhu, “Learning Entity and Relation Embeddings for Knowledge Graph Completion,” *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 29, no. 1, Feb. 2015. [Online]. Available: <https://ojs.aaai.org/index.php/AAAI/article/view/9491>

- [66] D. Liu, “A Comparison of Zero/Few-shot Approaches to Natural Language Understanding,” Ph.D. dissertation, 2023. [Online]. Available: [https://isl.anthropomatik.kit.edu/downloads/MA\\_Danqing\\_Liu.pdf](https://isl.anthropomatik.kit.edu/downloads/MA_Danqing_Liu.pdf)
- [67] W. Liu, P. Zhou, Z. Zhao, Z. Wang, Q. Ju, H. Deng, and P. Wang, “K-BERT: Enabling Language Representation with Knowledge Graph,” *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 03, pp. 2901–2908, Apr. 2020. [Online]. Available: <https://ojs.aaai.org/index.php/AAAI/article/view/5681>
- [68] X. Liu, Q. Tong, X. Liu, and Z. Qin, “Ontology Matching: State of the Art, Future Challenges, and Thinking Based on Utilized Information,” *IEEE Access*, vol. 9, pp. 91 235–91 243, 2021, conference Name: IEEE Access. [Online]. Available: <https://ieeexplore.ieee.org/document/9347437>
- [69] Y. Liu, W. Che, Y. Wang, B. Zheng, B. Qin, and T. Liu, “Deep Contextualized Word Embeddings for Universal Dependency Parsing,” *ACM Transactions on Asian and Low-Resource Language Information Processing*, vol. 19, no. 1, pp. 1–17, Jan. 2020. [Online]. Available: <https://dl.acm.org/doi/10.1145/3326497>
- [70] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, “RoBERTa: A Robustly Optimized BERT Pretraining Approach,” *arXiv e-prints*, p. arXiv:1907.11692, Jul. 2019, \_eprint: 1907.11692. [Online]. Available: <https://arxiv.org/abs/1907.11692>
- [71] J. López Espejel, E. H. Ettifouri, M. S. Yahaya Alassan, E. M. Chouham, and W. Dahhane, “GPT-3.5, GPT-4, or BARD? Evaluating LLMs reasoning ability in zero-shot setting and performance boosting through prompts,” *Natural Language Processing Journal*, vol. 5, p. 100032, Dec. 2023. [Online]. Available: <https://linkinghub.elsevier.com/retrieve/pii/S2949719123000298>
- [72] J. Madhavan, P. Bernstein, AnHai Doan, and A. Halevy, “Corpus-Based Schema Matching,” in *21st International Conference on Data Engineering (ICDE’05)*. Tokyo, Japan: IEEE, 2005, pp. 57–68. [Online]. Available: <http://ieeexplore.ieee.org/document/1410106/>
- [73] J. Madhavan, P. A. Bernstein, and E. Rahm, “Generic Schema Matching with Cupid,” in *Proceedings of the 27th International Conference on Very Large Data Bases*, ser. VLDB ’01. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2001, pp. 49–58. [Online]. Available: <https://dl.acm.org/doi/abs/10.5555/645927.672191>
- [74] C. W. F. Mayer, S. Ludwig, and S. Brandt, “Prompt text classifications with transformer models! An exemplary introduction to prompt-based learning with large language models,” *Journal of Research on Technology in Education*, vol. 55, no. 1, pp. 125–141, Jan. 2023, publisher: Routledge \_eprint: <https://doi.org/10.1080/15391523.2022.2142872>. [Online]. Available: <https://doi.org/10.1080/15391523.2022.2142872>
- [75] S. Melnik, H. Garcia-Molina, and E. Rahm, “Similarity flooding: a versatile graph matching algorithm and its application to schema matching,” in *Proceedings 18th International Conference on Data Engineering*. San Jose, CA, USA: IEEE Comput. Soc, 2002, pp. 117–128. [Online]. Available: <http://ieeexplore.ieee.org/document/994702/>
- [76] Y. Meng, M. Michalski, J. Huang, Y. Zhang, T. Abdelzaher, and J. Han, “Tuning Language Models as Training Data Generators for Augmentation-Enhanced Few-Shot Learning,” in *Proceedings of the 40th International Conference on Machine Learning*. PMLR, Jul. 2023, pp. 24 457–24 477, iSSN: 2640-3498. [Online]. Available: <https://proceedings.mlr.press/v202/meng23b.html>

- 
- [77] S. Mežnar, M. Bevec, N. Lavrač, and B. Škrlič, “Ontology Completion with Graph-Based Machine Learning: A Comprehensive Evaluation,” *Machine Learning and Knowledge Extraction*, vol. 4, no. 4, pp. 1107–1123, Dec. 2022. [Online]. Available: <https://www.mdpi.com/2504-4990/4/4/56>
- [78] T. Mikolov, K. Chen, G. Corrado, and J. Dean, “Efficient Estimation of Word Representations in Vector Space,” *Proceedings of Workshop at ICLR*, vol. 2013, Jan. 2013. [Online]. Available: <https://arxiv.org/abs/1301.3781>
- [79] A. Miles and J. R. Pérez-Agüera, “SKOS: Simple Knowledge Organisation for the Web,” 2007. [Online]. Available: <https://www.w3.org/TR/skos-reference/>
- [80] G. A. Miller, “WordNet: a lexical database for English,” *Communications of the ACM*, vol. 38, no. 11, pp. 39–41, Nov. 1995. [Online]. Available: <https://dl.acm.org/doi/10.1145/219717.219748>
- [81] S. Neutel and M. De Boer, “Towards Automatic Ontology Alignment using BERT.” 2021. [Online]. Available: <https://ceur-ws.org/Vol-2846/paper28.pdf>
- [82] M. Nickel, V. Tresp, and H.-P. Kriegel, “A three-way model for collective learning on multi-relational data,” in *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ser. ICML’11. Madison, WI, USA: Omnipress, 2011, pp. 809–816, event-place: Bellevue, Washington, USA. [Online]. Available: <https://dl.acm.org/doi/10.5555/3104482.3104584>
- [83] I. Nkisi-Orji, N. Wiratunga, S. Massie, K.-Y. Hui, and R. Heaven, “Ontology Alignment Based on Word Embedding and Random Forest Classification,” in *Machine Learning and Knowledge Discovery in Databases*, M. Berlingerio, F. Bonchi, T. Gärtner, N. Hurley, and G. Ifrim, Eds. Cham: Springer International Publishing, 2019, vol. 11051, pp. 557–572, series Title: Lecture Notes in Computer Science. [Online]. Available: [http://link.springer.com/10.1007/978-3-030-10925-7\\_34](http://link.springer.com/10.1007/978-3-030-10925-7_34)
- [84] H. Nori, N. King, S. M. McKinney, D. Carignan, and E. Horvitz, “Capabilities of GPT-4 on Medical Challenge Problems,” Apr. 2023, arXiv:2303.13375 [cs]. [Online]. Available: <http://arxiv.org/abs/2303.13375>
- [85] I. Osman, S. B. Yahia, and G. Diallo, “Ontology Integration: Approaches and Challenging Issues,” *Information Fusion*, vol. 71, pp. 38–63, 2021. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S1566253521000130>
- [86] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, “Deep Contextualized Word Representations,” in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. New Orleans, Louisiana: Association for Computational Linguistics, Jun. 2018, pp. 2227–2237. [Online]. Available: <https://aclanthology.org/N18-1202>
- [87] M. E. Peters, M. Neumann, R. Logan, R. Schwartz, V. Joshi, S. Singh, and N. A. Smith, “Knowledge Enhanced Contextual Word Representations,” in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, K. Inui, J. Jiang, V. Ng, and X. Wan, Eds. Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 43–54. [Online]. Available: <https://aclanthology.org/D19-1005>

- 
- [88] J. B. Pollack, “Recursive distributed representations,” *Artificial Intelligence*, vol. 46, no. 1, pp. 77–105, 1990. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/000437029090005K>
- [89] J. Portisch, M. Hladik, and H. Paulheim, “Background Knowledge in Ontology Matching: A Survey,” 2022. [Online]. Available: <https://www.semantic-web-journal.net/content/background-knowledge-ontology-matching-survey-0>
- [90] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, “Language Models are Unsupervised Multitask Learners,” 2019. [Online]. Available: [https://d4mucfpksywv.cloudfront.net/better-language-models/language\\_models\\_are\\_unsupervised\\_multitask\\_learners.pdf](https://d4mucfpksywv.cloudfront.net/better-language-models/language_models_are_unsupervised_multitask_learners.pdf)
- [91] E. Rahm and P. A. Bernstein, “A survey of approaches to automatic schema matching,” *The VLDB Journal*, vol. 10, no. 4, pp. 334–350, Dec. 2001. [Online]. Available: <https://doi.org/10.1007/s007780100057>
- [92] N. Reimers and I. Gurevych, “Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks,” in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, K. Inui, J. Jiang, V. Ng, and X. Wan, Eds. Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 3982–3992. [Online]. Available: <https://aclanthology.org/D19-1410>
- [93] M. S. M. Rudwan and J. V. Fonou-Dombeu, “Hybridizing Fuzzy String Matching and Machine Learning for Improved Ontology Alignment,” *Future Internet*, vol. 15, no. 7, p. 229, Jun. 2023. [Online]. Available: <https://www.mdpi.com/1999-5903/15/7/229>
- [94] S. Saki Norouzi, M. Mahdavinejad, and P. Hitzler, “Conversational Ontology Alignment with ChatGPT,” Aug. 2023. [Online]. Available: <https://arxiv.org/abs/2308.09217>
- [95] G. Salton, A. Wong, and C. S. Yang, “A vector space model for automatic indexing,” *Communications of the ACM*, vol. 18, no. 11, pp. 613–620, Nov. 1975. [Online]. Available: <https://dl.acm.org/doi/10.1145/361219.361220>
- [96] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, “DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter,” Feb. 2020, arXiv:1910.01108 [cs]. [Online]. Available: <http://arxiv.org/abs/1910.01108>
- [97] V. Schickel-Zuber and B. Faltings, “OSS: a semantic similarity function based on hierarchical ontologies,” in *Proceedings of the 20th International Joint Conference on Artificial Intelligence*, ser. IJCAI’07. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2007, pp. 551–556, event-place: Hyderabad, India. [Online]. Available: <https://www.ijcai.org/Proceedings/07/Papers/087.pdf>
- [98] K. . Service, “70% van krapteberoepen kent overschot in ander Europees land,” Jun. 2023, last Modified: 2023-06-12. [Online]. Available: <https://www.uwv.nl/nl/persberichten/70-procent-van-krapteberoepen-kent-overschot-in-ander-europees-land>
- [99] R. Shaan, “A Comprehensive Review of Classic and Modern Techniques for Ontology Matching,” Jun. 2023. [Online]. Available: <https://hal.science/hal-04162262>
- [100] A. Singhal, “Introducing the knowledge graph: Things, not strings,” May 2012, publication Title: Google. [Online]. Available: <https://blog.google/products/search/introducing-knowledge-graph-things-not/>

- [101] V. Spiliopoulos, G. A. Vouros, and V. Karkaletsis, “On the discovery of subsumption relations for the alignment of ontologies,” *Journal of Web Semantics*, vol. 8, no. 1, pp. 69–88, Mar. 2010. [Online]. Available: <https://linkinghub.elsevier.com/retrieve/pii/S1570826810000028>
- [102] C. Sun, X. Qiu, Y. Xu, and X. Huang, “How to Fine-Tune BERT for Text Classification?” in *Chinese Computational Linguistics*, ser. Lecture Notes in Computer Science, M. Sun, X. Huang, H. Ji, Z. Liu, and Y. Liu, Eds. Cham: Springer International Publishing, 2019, pp. 194–206. [Online]. Available: [https://link.springer.com/chapter/10.1007/978-3-030-32381-3\\_16](https://link.springer.com/chapter/10.1007/978-3-030-32381-3_16)
- [103] X. Sun, X. Li, J. Li, F. Wu, S. Guo, T. Zhang, and G. Wang, “Text Classification via Large Language Models,” in *Findings of the Association for Computational Linguistics: EMNLP 2023*, H. Bouamor, J. Pino, and K. Bali, Eds. Singapore: Association for Computational Linguistics, Dec. 2023, pp. 8990–9005. [Online]. Available: <https://aclanthology.org/2023.findings-emnlp.603>
- [104] M. Tounsi Dhouib, C. Faron, and A. G. B. Tettamanzi, “Measuring Clusters of Labels in an Embedding Space to Refine Relations in Ontology Alignment,” *Journal on Data Semantics*, vol. 10, no. 3-4, pp. 399–408, Dec. 2021. [Online]. Available: <https://link.springer.com/10.1007/s13740-021-00137-8>
- [105] T. Trouillon, J. Welbl, S. Riedel, E. Gaussier, and G. Bouchard, “Complex Embeddings for Simple Link Prediction,” in *Proceedings of The 33rd International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, M. F. Balcan and K. Q. Weinberger, Eds., vol. 48. New York, New York, USA: PMLR, Jun. 2016, pp. 2071–2080. [Online]. Available: <https://proceedings.mlr.press/v48/trouillon16.html>
- [106] N. A. Vasilevsky, N. A. Matentzoglou, S. Toro, J. E. F. IV, H. Hegde, D. R. Unni, G. F. Alyea, J. S. Amberger, L. Babb, J. P. Balhoff, T. I. Bingaman, G. A. Burns, O. J. Buske, T. J. Callahan, L. C. Carmody, P. C. Cordo, L. E. Chan, G. S. Chang, S. L. Christiaens, L. C. Daugherty, M. Dumontier, L. E. Failla, M. J. Flowers, J. H. Alpha Garrett, J. L. Goldstein, D. Gratton, T. Groza, M. Hanauer, N. L. Harris, J. A. Hilton, D. S. Himmelstein, C. T. Hoyt, M. S. Kane, S. Köhler, D. Lagorce, A. Lai, M. Larralde, A. Lock, I. L. Santiago, D. R. Maglott, A. J. Malheiro, B. H. M. Meldal, M. C. Munoz-Torres, T. H. Nelson, F. W. Nicholas, D. Ochoa, D. P. Olson, T. I. Oprea, D. Osumi-Sutherland, H. Parkinson, Z. M. Pendlington, A. Rath, H. L. Rehm, L. Remennik, E. R. Riggs, P. Roncaglia, J. E. Ross, M. F. Shadbolt, K. A. Shefchek, M. N. Similuk, N. Sioutos, D. Smedley, R. Sparks, R. Stefancsik, R. Stephan, A. L. Storm, D. Stupp, G. S. Stupp, J. C. Sundaramurthi, I. Tammen, D. Tay, C. L. Thaxton, E. Valasek, J. Valls-Margarit, A. H. Wagner, D. Welter, P. L. Whetzel, L. L. Whiteman, V. Wood, C. H. Xu, A. Zankl, X. A. Zhang, C. G. Chute, P. N. Robinson, C. J. Mungall, A. Hamosh, and M. A. Haendel, “Mondo: Unifying diseases for the world, by the world,” *medRxiv*, 2022, publisher: Cold Spring Harbor Laboratory Press \_eprint: <https://www.medrxiv.org/content/early/2022/05/03/2022.04.13.22273750.full.pdf>. [Online]. Available: <https://www.medrxiv.org/content/early/2022/05/03/2022.04.13.22273750>
- [107] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Kaiser, and I. Polosukhin, “Attention is All you Need,” in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates, Inc., 2017. [Online]. Available: [https://proceedings.neurips.cc/paper\\_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf](https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf)

- [108] R. Vieira and K. Revoredo, “Using word semantics on entity names for correspondence set generation.” in *OM@ ISWC*, 2017, pp. 223–224. [Online]. Available: [https://ceur-ws.org/Vol-2032/om2017\\_poster9.pdf](https://ceur-ws.org/Vol-2032/om2017_poster9.pdf)
- [109] H. Wache, T. Vögele, U. Visser, H. Stuckenschmidt, G. Schuster, H. Neumann, and S. Ubner, “Ontology-Based Integration of Information - A Survey of Existing Approaches,” *Proceedings of the IJCAI’01 Workshop on Ontologies and Information Sharing, Seattle, Washington, USA, Aug 4-5*, Aug. 2002. [Online]. Available: <https://ceur-ws.org/Vol-47/wache.pdf>
- [110] L. L. Wang, C. Bhagavatula, M. Neumann, K. Lo, C. Wilhelm, and W. Ammar, “Ontology alignment in the biomedical domain using entity definitions and context,” in *Proceedings of the BioNLP 2018 workshop*, D. Demner-Fushman, K. B. Cohen, S. Ananiadou, and J. Tsujii, Eds. Melbourne, Australia: Association for Computational Linguistics, Jul. 2018, pp. 47–55. [Online]. Available: <https://aclanthology.org/W18-2306>
- [111] Z. Wang, J. Zhang, J. Feng, and Z. Chen, “Knowledge Graph Embedding by Translating on Hyperplanes,” *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 28, no. 1, Jun. 2014. [Online]. Available: <https://ojs.aaai.org/index.php/AAAI/article/view/8870>
- [112] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. R. Salakhutdinov, and Q. V. Le, “XLNet: Generalized Autoregressive Pretraining for Language Understanding,” in *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d. Alché-Buc, E. Fox, and R. Garnett, Eds., vol. 32. Curran Associates, Inc., 2019. [Online]. Available: [https://proceedings.neurips.cc/paper\\_files/paper/2019/file/dc6a7e655d7e5840e66733e9ee67cc69-Paper.pdf](https://proceedings.neurips.cc/paper_files/paper/2019/file/dc6a7e655d7e5840e66733e9ee67cc69-Paper.pdf)
- [113] L. Yao, C. Mao, and Y. Luo, “KG-BERT: BERT for Knowledge Graph Completion,” Sep. 2019, arXiv:1909.03193 [cs]. [Online]. Available: <http://arxiv.org/abs/1909.03193>
- [114] Q. Zeng, W. Yu, M. Yu, T. Jiang, T. Weninger, and M. Jiang, “Tri-Train: Automatic Pre-Fine Tuning between Pre-Training and Fine-Tuning for SciNER,” in *Findings of the Association for Computational Linguistics: EMNLP 2020*, T. Cohn, Y. He, and Y. Liu, Eds. Online: Association for Computational Linguistics, Nov. 2020, pp. 4778–4787. [Online]. Available: <https://aclanthology.org/2020.findings-emnlp.429>
- [115] Y. Zhang, X. Wang, S. Lai, S. He, K. Liu, J. Zhao, and X. Lv, “Ontology Matching with Word Embeddings,” in *Chinese Computational Linguistics and Natural Language Processing Based on Naturally Annotated Big Data*, M. Sun, Y. Liu, and J. Zhao, Eds. Cham: Springer International Publishing, 2014, vol. 8801, pp. 34–45, series Title: Lecture Notes in Computer Science. [Online]. Available: [http://link.springer.com/10.1007/978-3-319-12277-9\\_4](http://link.springer.com/10.1007/978-3-319-12277-9_4)
- [116] Z. Zhang, X. Han, Z. Liu, X. Jiang, M. Sun, and Q. Liu, “ERNIE: Enhanced Language Representation with Informative Entities,” in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, A. Korhonen, D. Traum, and L. Màrquez, Eds. Florence, Italy: Association for Computational Linguistics, Jul. 2019, pp. 1441–1451. [Online]. Available: <https://aclanthology.org/P19-1139>
- [117] N. Zong, S. Nam, J.-H. Eom, J. Ahn, H. Joe, and H.-G. Kim, “Aligning ontologies with subsumption and equivalence relations in Linked Data,” *Knowledge-Based Systems*, vol. 76, pp. 30–41, Mar. 2015. [Online]. Available: <https://linkinghub.elsevier.com/retrieve/pii/S0950705114004225>

# Appendix A

## Full experiments table

	Random	STROMA	TaSeR base	TaSeR	S-TaSeR	GPT4 zS	GPT4 FS
W1	0.33 (-)	0.33 (0.1)	0.21 (0.4)	0.44 (2.2)	0.43 (2.2)	0.51 (4.8)	0.39 (5.7)
W2	0.33 (-)	0.41 (0.1)	0.29 (0.4)	0.46 (0.5)	0.39 (0.5)	0.42 (1.5)	0.43 (1.8)
W3	0.33 (-)	0.39 (0.1)	0.25 (0.4)	0.80 (0.6)	0.62 (0.6)	0.77 (2.1)	0.52 (2.5)
W4	0.33 (-)	0.56 (0.1)	0.30 (0.5)	0.83 (0.6)	0.47 (0.5)	0.76 (1.8)	0.62 (2.2)
L1	0.33 (-)	0.11 (0.9)	0.23 (0.5)	0.48 (2.6)	0.48 (2.7)	0.54 (74.8)	0.45 (88.5)
L2	0.33 (-)	0.18 (0.5)	0.31 (0.4)	0.45 (1.8)	0.40 (1.9)	0.47 (38.0)	0.40 (44.9)
B1	0.33 (-)	0.24 (0.9)	0.29 (0.5)	0.90 (5.3)	0.90 (5.6)	0.68 (68.3)	0.73 (80.8)
B2	0.33 (-)	0.07 (0.5)	0.10 (0.4)	0.58 (4.7)	0.58 (5.0)	0.42 (37.5)	0.31 (44.4)
B3	0.33 (-)	0.00 (0.9)	0.17 (0.6)	0.34 (27.5)	0.35 (29.4)	0.61 (68.3)	0.59 (80.8)
B4	0.33 (-)	0.02 (0.5)	0.09 (0.4)	0.82 (5.3)	0.79 (5.5)	0.57 (40.5)	0.42 (47.9)
B5	0.33 (-)	0.00 (0.9)	0.26 (0.5)	0.85 (7.3)	0.82 (7.6)	0.57 (68.3)	0.55 (80.8)

Table A.1: Macro average F1 score per method per dataset, supplemented with the runtime in minutes