Czech Technical University in Prague

Faculty of Electrical Engineering
Department of Computer Science

Information Extraction and Ontology Learning from Text with Limited Resources

Doctoral Thesis

Ing. Lama Saeeda

Prague, May 2023

Ph.D. Programme: Electrical Engineering and Information Technology
Branch of study: Artificial Intelligence and Biocybernetics

Supervisor: Ing. Petr Křemen, Ph.D.

Thesis Supervisor:
Ing. Petr Křemen, Ph.D.
Department of Computer Science and Engineering
Faculty of Electrical Engineering
Czech Technical University in Prague
Technická 2
160 00 Prague 6
Czech Republic

Declaration

I hereby declare I have written this doctoral thesis independently and quoted all the sources of information used in accordance with methodological instructions on ethical principles for writing an academic thesis. Moreover, I state that this thesis has neither been submitted nor accepted for any other degree.

In Prague, May 2023

.....
Ing. Lama Saeeda

Abstract

Ontology-based information extraction from unstructured textual documents has emerged as an extension to the classic field of information extraction, where background knowledge is a first-class citizen in the extraction pipeline. In addition, ontology learning from a text written in natural language is a well-studied domain. However, the applicability of techniques for extracting semantic entities and learning ontology from natural language texts is strongly dependent on the characteristics of the text corpus and the language used. In this thesis, we investigate the available information extraction and entity-linking methods and tools. We discuss the limitation and challenges of applying these methods on a low-resource document corpus. Then, we introduce an end-to-end approach to entity linking and learning new ontological entities from a corpus with limited resources. We present an entity linking method to provide links between the ontology and the text for contexts where machine learning-based methods are difficult to apply. We prototype the method for Czech resources that can be easily adapted by other languages. Then, we discuss the benefits of adequately designed Lexico-Semantic patterns in ontology learning. We propose a preliminary set of Lexico-Semantic patterns designed for the Czech language to learn new relations between concepts in the related ontology. We introduce the Temporal Descriptor ontology that can be extended to enhance the ontology learning process with the temporal dimension, and finally, we present the applicability of the approach to real-world applications.

Keywords: Entity Linking, Ontology Learning, Ontology-Based Information Extraction, Lexico-Semantic Patterns, Semantic Web, Natural Language Processing, Temporal Descriptor.

Abstrakt

Extrakce informací z nestrukturovaných textových dokumentů za využití ontologií umožňuje oproti klasickým metodám extrakce informací využít explicitní znalosti uložené v ontologiích ke zvýšení kvality extrakce. Tvorba ontologie z textu napsaném v přirozeném jazyce je sice velmi dobře prozkoumanou oblastí, nicméně využitelnost stávajících technik pro extrakci sémantických entit a semi-automatické tvorby ontologie z textů je plně závislá na charakteristikách textového korpusu a na použitém jazyce. V této práci zkoumám dostupné metody extrakce informací a metody propojování entit a nástroje s nimi spojené zejména s ohledem na omezení těchto technik pro práci s jazyky s omezenými zdroji. Dále představuji holistický přístup k propojování entit, fungující i v případech, kdy jsou špatně aplikovatelné metody založené na strojovém učení. V práci představuji prototyp metody pro české zdroje, který může být zároveň jednoduše adaptovatelný i pro další jazyky. Dále diskutuji výhody adekvátně navržených lexikálně-sémantických vzorů pro tvorbu ontologie a navrhuji jejich sadu pro český jazyk, které dokážou vytvářet nové vztahy mezi koncepty v dané ontologii. Posledním příspěvkem má práce je ontologie časového deskriptoru (Temporal Descriptor),využitelná k vylepšení procesu tvorby ontologie s časovou dimenzí. V závěru prezentuji aplikace vytvořených metod v praktických aplikacích v oblasti územního plánovaná a letecké bezpečnosti.

Klíčová slova: Spojování entit, Tvorba ontologie, Extrakce informací za využití ontologie, Lexikálně-sémantické vzory, Sémantické sítě, Zpracování přirozeného jazyka, Temporal Descriptor.

Acknowledgements

Acknowledgment:
I am deeply grateful to my thesis supervisor, Dr. Petr Křemen. His invaluable guidance, encouragement, and profound knowledge have been instrumental in shaping this research.

My sincere thanks to the Department of Computer Science and my colleagues in the Knowledge-Based and Software Systems group for their support, resources, opportunities, and collaborations that have enriched my research journey.

To my family and friends, your countless words of heartening and understanding during the highs and lows have been a constant source of strength. Your unwavering support, love, and patience mean the world to me.

Dedication:
To my loving parents, who have always believed in me and supported my journey.

List of Tables

List of Figures

List of Acronyms

AAII Air Accidents Investigation Institute.

ASO Aviation Safety Ontology.

DOLCE Descriptive Ontology for Linguistic and Cognitive Engineering.

EL Entity Linking.

GATE General Architecture for Text Engineering.

GPT-3 Generative Pre-trained Transformer 3.

HIEE HIEL Information Extraction Engine.

HIEL Hermes Information Extraction Language.

HNP Hermes News Portal.

IE Information Extraction.

JAPE Java Annotation Patterns Engine.

LHS Left-Hand Side.

LLMs Large Language Models.

LOD Linked Open Data.

LOV Linked Open Vocabularies.

LRL Limited-Resources Language.

LSPs Lexico-Semantic Patterns.

MPP Metropolitan Plan of Prague.

NER Named Entity Recognition.

NLP Natural Language Processing.

OBIE Ontology-Based Information Extraction.

ODPs Ontology Design Patterns.

OL Ontology Learning.

OWL Web Ontology Language.

POS Part of Speech.

RDF Resource Description Framework.

RHS Right-Hand Side.

SABiO Systematic Approach for Building Ontologies.

TDO Temporal Descriptor Ontology.

TF-IDF Term Frequency - Inverse Document Frequency.

UFO Unified Foundation Ontology.

Contents

Chapter 1

Introduction

Semantic technologies provide advancement in information systems by assigning semantics to data by means of a shared formal ontology. The ontology is especially useful because it supports the exchange and sharing of information, as well as reasoning tasks, allowing systems to automatically infer new knowledge based on the concepts, relationships, and axioms defined in the ontology. Moreover, ontology is essential in any studied domain, for example, biomedical research, aviation industry, urban planning, and development projects, etc. to perform important tasks, such as harmonizing data capture, supporting various Natural Language Processing (NLP) and Information Extraction (IE) tasks, and providing a common understanding of the technical terms used in the domain, facilitating effective knowledge management and communication. For example, in urban planning and development, a master plan is a legal tool for global planning that aims to support the urban character of the various localities. It addresses the future of the city, including the development of infrastructure and areas for new construction. Different regulations can apply to different parts of the plan, for example, building regulations. Also, it involves many actors in building and developing the plan, including urban planning experts, inhabitants, experts from the legal and law department, and even politicians. Communication between all of these parties is not an easy process and involves a wide range of technical terms and ambiguous jargon. For this reason, it is crucial to normalize an efficient way of communication through, e.g., an urban planning ontology that allows a common understanding of the technical terms and the relations between these terms that might cause confusion among all participants. In addition, including temporal and geographical dimensions in such ontology can be utilized in various ways, such as capturing the evolution of urban systems over time, including infrastructure, demographics, and changes in land use, such as the conversion of agricultural land to residential or commercial use.

However, using such an ontology depends directly on the availability of this

ontology in the target domain. Building a domain ontology based on a set of unstructured documents manually is tremendously exhaustive in terms of time and effort expended by human experts. Usually, domain experts, besides knowledge engineers, spend a lot of time reviewing available textual resources and documents in order to build a background knowledge that supports the studied domain. This is slow and expensive, especially for a large volume of documents. In addition, it is prone to human errors and might suffer from inconsistencies and subjectivity. This process can be enhanced by utilizing natural language processing side-by-side with information extraction techniques to help develop the ontology. Automatic information extraction has significantly improved the accuracy, efficiency, scalability, and adaptability of the extraction process.

Ontology Learning (OL) from a textual corpus is the set of methods and techniques used to build an ontology from scratch, enrich, or adapt an existing ontology in a semi-automatic fashion using several knowledge and information sources [1]. These techniques are divided into two main types, rule-based and machine learning approaches.

Machine learning approaches can be very effective for various ontology learning tasks, but they suffer from many disadvantages when applied to low-resource domain-specific textual documents. A low-resource domain is the area of knowledge for which there is limited availability of textual data or sufficient annotated corpora, or even written in a low-resource language that lacks the proper processing tools, such as Slavic languages, compared to rich, well-studied mainstream ones, for example, English.

We highlight the main disadvantages and limitations as follows:

- Limited training data - Machine learning algorithms require a vast amount of annotated training data to learn to recognize entities and relationships. However, for low-resource languages, limited annotated data can be available to train machine learning models, which can limit their effectiveness.

- Limited-Resources Language (LRL) - some languages may have limited language resources available, such as dictionaries, ontologies, or Named Entity Recognition (NER) tools. This can make it more difficult to develop and evaluate machine learning models for these languages and to ensure the accuracy of the resulting ontology.

- Limited portability - Machine learning models trained on one domain (Aviation safety, Urban planning, Time expressions, etc.) may not generalize well to other domains or topics due to differences in the vocabulary and language used.

- Ambiguity and complex morphology - Some languages have complex morphology. This complexity can make it difficult for models to extract entities and relationships from the text. For example, in the Czech language, nouns, verbs, and adjectives can have many different forms depending on their grammatical roles in a sentence. This can make it difficult to accurately disambiguate entities and relationships and can lead to errors in the ontology.

Generally, the main challenges of deep learning methods in ontology learning are to develop appropriate training datasets and define suitable modeling of the problem, besides deciding what the input and the output of the deep network should be [2]. Similarly, Large Language Models (LLMs) such as Generative Pre-trained Transformer 3 (GPT-3) [3] and its more recent versions are very promising in many applications and can provide initial results to extract entities and relations from the text, mainly by reframing the ontology learning task into a prompt completion task as in [4]. However, the aforementioned challenges still persist for languages other than English and domains with limited learning resources. Furthermore, the generation of incorrect statements, that is, hallucinations [5], and false positives, are additional challenges that could lead to inaccurate ontology construction or extraction of incorrect information (which even does not actually exist in the text or domain). This can have a huge impact on the accuracy and reliability of the resulting ontology. Further research efforts may be needed to fine-tune LLMs, or even combine them with other techniques to overcome these challenges.

To address these challenges when building a domain ontology based on a set of unstructured documents written in a low-resource language, it may be necessary to develop a language-specific approach that takes into account the unique features of the language, such as rule-based methods or hybrid approaches that combine machine learning with linguistic knowledge. In addition, rule-based approaches can be used for domain-specific tasks, as they can be customized to the specific linguistic patterns and vocabulary used in the domain.

Building a domain ontology based on unstructured documents is not a simple task and might require combining multiple techniques. To construct an ontology in a studied domain, a seed ontology can be created based on available public standard ontologies, controlled vocabularies, internal lists of terms or keywords, etc.; the seed ontology can then be used to automatically annotate available documents, standard operating procedures, manuals, best-practice documents, etc. On the other hand, the annotations in the text can be used to extract entities and relationships by combining existing linguistic and semantic knowledge and augmenting the ontology with additional knowledge. We view Information Extraction and Ontology Learning as two sides of the same coin, where the

ontology can evolve further with the augmented entities and relations extracted from the text, and the extracted entities from the text can be highly affected by the evolved background knowledge. Hence, OL is an accumulative process and evolves with more executions of the two tasks.

In this thesis, we introduce a method for the continuous development of domain ontology, based on a seed ontology and an extended set of domain-related documents. The approach consists of two phases that are executed in an iterative fashion. Entity linking phase and ontology learning phase.

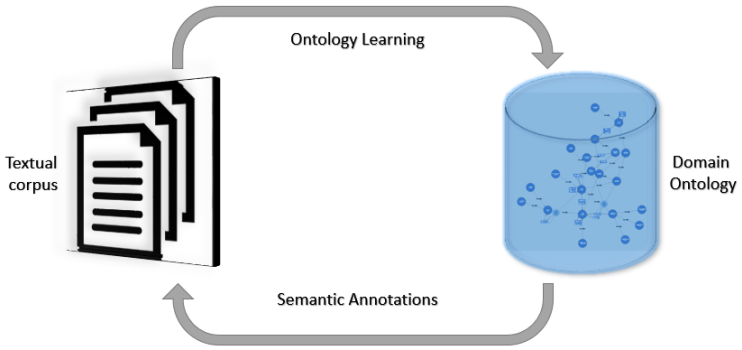


Figure 1.1: Information Extraction and Ontology Learning Iterative Approach

To illustrate our approach, consider the following example taken from an urban planning and development document in Czech.

Cs: "Správní území Prahy členěno na lokality"
En: "Administrative territory of Prague divided into localities"

At first, the entity linking engine enhances the text with semantic information by providing links to the terms in the ontology.

Cs: "**Správní území Prahy** členěno na **lokality**" Where:
Správní území Prahy is linked to mpp[1]:správní-území-prahy and
lokality is linked to mpp:lokalita

Using this information with a well-designed rule reveals the relation between concepts.

mpp:správní-území-prahy *hasPart* **mpp:lokalita**

This revealed relation then can be suggested to the user to be added to the ontology.

The ontology learning approach consists of the following two phases:

[1]mpp: http://onto.fel.cvut.cz/ontologies/slovnik/datovy-mpp-3.5-np/pojem/

- Document processing and entity linking task: This step enhances documents with semantic and syntactic information. It provides links between textual documents and the concepts that are defined in the seed ontology to add a semantic context to the processed documents.

- Learning ontological entities task: In this step, a set of rule-based Lexico-Semantic Patterns (LSPs) is applied to the contextualized text to enhance the process of learning new entities and relations between concepts.

The iterative approach suggests having a seed ontology that can be acquired in an additional knowledge acquisition step that proceeds the two phases. Experts can also revise each phase's output to ensure the quality of the resulting ontology.

1.1 Contribution

- An end-to-end iterative methodology for information extraction and ontology learning from unstructured text.

- A vanilla approach to entity linking for contexts where machine learning-based methods are difficult to apply. A prototype of an entity linking tool for the Czech language, Annotace, and its integration into the vocabulary management tool TermIt.

- A semantic rule-based approach to enhance the ontology with new entities and relations for these contexts. We introduce a preliminary, extendable set of Lexico-Semantic Patterns to learn new ontological entities.

- Evaluation of existing entity linking tools on real-world data.

- Temporal descriptor ontology that can be extended to enhance the ontology learning process with the temporal dimension.

1.2 Structure of the text

The rest of the thesis is organized as follows, Chapter 2 describes the state-of-the-art techniques and a general background of related topics. Chapter 3 explains the proposed methodology in further detail. In Chapters 4 and 5, we present our work in the field of entity linking and ontology learning respectively. In Chapter 6 we show multiple use cases in which the techniques are used in real-world applications, and Chapter 7 summarizes the thesis and suggests further steps and recommendations.

Chapter 2

State of The Art

This chapter is divided into two sections. The first section presents the necessary technologies, techniques, and tools required to comprehend the subsequent chapters. The second section outlines the related work.

2.1 Background

The aim of this chapter is to provide an overview of the technologies used in our pipeline for Ontology-Based Information Extraction (OBIE) and Ontology Learning (OL).

2.1.1 Ontology

Originally, the term ontology is a philosophical discipline concerned with the study of the nature of being and existence. Today, the most frequently quoted definition of ontology in the computer science literature is Gruber's "An ontology is a formal, explicit specification of a shared conceptualization" [6]. Ontology is a formal representation of knowledge by a set of concepts within a domain and the relationships between those concepts. The W3C consortium suggests that the ontology should provide descriptions for classes (or 'things'), relationships among those classes, and properties that the classes should have.

Ontology has varying degrees of expressiveness. In lightweight ontology, concepts are connected by rather general associations than strict formal connections, while formal ontology makes intensive use of axioms for specification.

Capturing knowledge is the key to building powerful and large AI systems. Thus, ontology has been used in many areas of computer science, such as Artificial Intelligence, Natural Language Processing, and Software Engineering, to enable the analysis and reuse of domain knowledge, limit complexity, and organize information.

2.1.1.1 Domain Ontology

Ontology forms the heart of any system of knowledge representation for any given domain. It is usually restricted to a specific application area to be manageable. A domain ontology is a representation of some part of reality, e.g. medicine, safety, physics, etc. Domain ontology is often developed to describe concepts, relationships, and other entities within a specific domain or subject area. It provides a standardized terminology and a set of rules that defines the meaning of terms and how they are related within the domain, which mitigates misunderstandings about terms in the field.

Domain ontologies are being developed for many fields such as healthcare, finance, engineering, etc., where a common understanding of concepts and their relationships is essential for effective communication and decision-making. In addition, they can be used to support a wide variety of applications, including information retrieval, knowledge management, decision support systems, and more.

On the other hand, multidomain ontology, like DBpedia, [7] covers multiple domains and contains a lot of instances, making it less formally structured and the data quality is lower with many inconsistencies.

2.1.1.2 Foundational Ontology

When building a new ontology, it is usually possible to reuse some existing ontology or other resources. Top-level ontologies, also called Foundational Ontologies, can be used, which describe the most general entities, contain generic specifications, and serve as a foundation for specializations. It allows specific ontologies that are built on top of it to share a common meta-model of basic concepts and relationships. This provides the ability for ontology merging and alignment methods to be applied. It also helps to understand the domain by checking how the entities relate to the generic model.

There are many well-known foundational ontologies; for example, the Descriptive Ontology for Linguistic and Cognitive Engineering (DOLCE) [8] that aims to capture the ontological categories underlying natural language and human common sense. Unified Foundation Ontology (UFO) [9] is another top-level ontology that has been developed with the special purpose of serving as a foundation for general conceptual modeling languages.

UFO draws inspiration from philosophy and modal logic and introduces class stereotypes such as Kind, Phase, Role, or Relator, and relationship stereotypes such as Mediation, Characterization, and several part-whole types.

The core category of UFO, the ontology of endurants **UFO-A**, has been employed to analyze structural conceptual modeling constructs such as object types and taxonomic relations, part-whole relations, roles, properties, and datatypes.

More recent developments incorporate an ontology of perdurants (Events, Processes) in UFO, **(UFO-B)** [10], which has been successfully employed as a reference model for addressing problems from complex media management, enterprise architecture, software engineering, and modeling of events in petroleum exploration [11].

2.1.1.3 OWL-Time ontology

The OWL-Time ontology[1] [12] is an OWL-2 DL ontology that provides a vocabulary for expressing facts about topological relations among instants and intervals, together with information about durations, and temporal positions, including date-time information.

The full definition of the OWL-Time Ontology can be found in [12]. Here, we present those parts that are essential for capturing temporal information found in the datasets. The basic structure of the ontology is based on an algebra of binary relations on intervals developed by Allen and Ferguson [13]. Temporal knowledge is represented by the class *time*[2]*:TemporalEntity*. This class has only two subclasses, *time:Instant* and *time:Interval* which axiomatize fundamental intuitions about time points (Instants) and time intervals (Intervals).

$$(\forall T)[TemporalEntity(T) \equiv Interval(T) \vee Instant(T)] \tag{2.1}$$

The OWL-Time ontology also offers the class *time:DateTimeInterval* which may be expressed using a single *time:DateTimeDescription* class. This class with its related properties is perfect for defining a higher level of granularity. For example, the day of the week, or a year, using the *time:TemporalUnit* class. *time:ProperInterval* class is used to represent proper intervals, which are intervals whose extremes are different.

2.1.2 NLP Concepts

NLP has been extensively studied by researchers. The primary distinction between NLP and IE is the objectives of both tasks. While the NLP task focuses on making sense of the text by determining its structure, sentiment, etc., the IE task aims to acquire concrete structured knowledge. Typically, IE tools and techniques often rely on NLP concepts. We highlight the main NLP concepts as follows.

Sentence splitting is the process of splitting text into individual sentences. This is important for sentence-level analysis, such as text summarization and relation extraction within the boundaries of a single sentence.

[1]https://www.w3.org/TR/owl-time/ Last accessed: 2023-05-29
[2]time: http://www.w3.org/2006/time#

Tokenization concerns breaking down the text into individual words or tokens. This allows further analysis of the text at the token level. Tokens can be words, numbers, or symbols.

Lemmatization is the technique to normalize the token into its base form called the lemma. The lemma corresponds to the canonical form of a real word. For example, the lemma of the token "Caring" is "Care". Lemmatization is used to reduce the number of unique words that need to be analyzed.

Stemming is similar to lemmatization but involves reducing the token to a simpler form, which does not necessarily correspond to a real word, but only to the fragment of the word that does not change in any state. For example, the stem of the token "Caring" is "Car".

Part of Speech (POS) Tagging is the process of identifying the part of speech of each token in the text. POS tagging is important for understanding the syntactic structure of a sentence, which is essential in entity recognition and extraction tasks.

2.1.3 Information Extraction from Text

Much of the world's knowledge is recorded in natural language text. Some of this knowledge has an implicit structure that needs to be revealed. This implicit structure can take the form of entities or relationships between the entities mentioned, which may not be easily discernible in the raw text. The goal of the Information Extraction (IE) task is to make the semantic structure of the text explicit to users, such as banks, business intelligence companies, or individual accountants, providing them with accurate and relevant information for informed decision-making, compliance, data analysis, automation, and information retrieval purposes, and to downstream applications that use this knowledge, such as text generation and summarization, and question-answering systems. Information extraction from the text also supports the building of domain ontologies. Many tools provide the functionality to extract information from textual documents both manually and automatically.

However, manual information extraction involves human annotators who manually identify and extract relevant information from text. This process is slow and expensive, especially for a large volume of documents. In addition, it is prone to human errors and might suffer from inconsistencies and subjectivity.

For these reasons, automated information extraction methods have attracted researchers to pull out the required information to perform many tasks, and they have used several techniques to achieve it. Automatic information extraction has significantly improved the accuracy, efficiency, scalability, and adaptability of the extraction process.

2.1.3.1 Ontology-Based Information Extraction

Information Extraction is the process of automatically obtaining knowledge from plain text. Because of the ambiguity of written natural language, Information Extraction is a difficult task. Ontology-Based Information Extraction reduces this complexity by including contextual information in the form of a domain ontology [1]. The approach starts from a particular document, or set of documents, and tries to identify entities found in that context trying to annotate them according to the input ontology. Contrary to plain IE systems, ontology-based ones are able to specify their output in terms of an existing ontology. Annotations represent a specific kind of metadata that provides references between entities appearing in resources and domain concepts modeled in an ontology. The information extraction task is usually divided into two main subtasks, preprocessing the textual documents and then performing chosen techniques to extract relevant information aligned with the knowledge base.

The goal of the preprocessing step is to perform modifications to the text in order to facilitate and improve the extraction process. Modifications can filter unwanted elements from the text (e.g., stopwords), enhance the text with new information (e.g., POS tagging), or transform the representation of the text into another representation (e.g., vector representation). These modifications are mostly independent of each other and are usually applied in a sequence. For example, it is very common that before a text is transformed into a vector representation, stop-words are filtered, and POS tagging is applied.

2.1.4 Ontology Learning

Using domain ontology for information extraction relies on the availability of this ontology in the domain of study. However, the automatic construction of the ontology is not trivial and requires lots of human intervention at some stages of the ontology construction. Ontology learning from text is the process of identifying terms, concepts, taxonomic relations, non-taxonomic relations, and axioms from textual information and using them to construct and maintain an ontology. Buitelar [14] presents the Ontology Learning Layer Cake for learning ontology. It divides the task into five sequential steps. Many methods followed this scenario to perform all the steps, while some performed only part of them. The output will be:

- Terms
- Concepts
- Taxonomic relations
- Non-taxonomic relations

Table 2.1: Ontology learning tasks and subtasks and the state-of-art techniques applied for each

		Terms	Concepts	Taxonomic R.	Non-taxonomic R.	Axioms
statistic methods	Text pre-processing	X				
	POS tagging	X				
	Sentence parsing	X				
	Latent semantic		X			
	Cooccurrence	X	X			
	Clustering		X	X		
	Term subsumption			X		
	Association rules					
Linguistic methods	Seed words	X				
	Semantic lexicon		X	X	X	
	Sub-categorization frames	X	X			
	Syntactic structure	X			X	
	Dependency analysis	X			X	
	Semantic templates			X	X	
	Lexico-syntactic paterns			X	X	
	Axiom templates					X
Logical methods	Logical inference			X	X	
	Inductive Logic					X

- Axioms

There are various approaches and tools available for the automatic construction of an ontology from a textual corpus, and they differ according to the problem they are aiming to solve.

The Ontology Learning task is divided into multiple subtasks, usually sequential. The main task is to extract the terms. Then, grouping the terms to form concepts. Finding the relations between these concepts is the next step, and finally generalizing the relations to form axioms.

The techniques employed by different systems may vary depending on the targeted tasks. Techniques can generally be classified into statistics-based, linguistics-based, logic-based, or hybrid [15]. The main state-of-the-art techniques, methods, and sources for each stage of the ontology learning process can be found in Table 2.1.

2.2 Related Work

This section provides an overview of related work in the area of Ontology-Based Information Extraction and Ontology Learning.

2.2.1 Ontology-Based Information Extraction

Information extractors from text can be implemented under two main strategies, extraction rules, as in [16], [17], and [18], or based on machine learning methods as in [19]. However, there are some IE systems that combine both implementation strategies into a hybrid extraction mechanism [20].

In [21], a survey of ontology-based approaches to semantic data extraction was performed, where researchers investigated why ontology has the potential to help semantic data extraction, and how formal semantics in the ontology can be incorporated into the data mining process, showing the advantages in

performing the data mining task that is not achievable with traditional data mining methods. A similar survey was conducted in [22] to provide an introduction to Ontology-Based Information Extraction and to review the details of different developed OBIE systems.

As discussed in [23], in order to discover new relationships between entities mentioned in the text, the extracted relation requires the process of mapping entities associated with the relation to the knowledge base before it could be populated into the knowledge base. The entity linking task is highly data dependent, and it is unlikely that a technique will dominate all others across all data sets [23]. The system requirements and the characteristics of the data sets affect the design of the entity-linking system.

Any entity linking system is usually based on two steps: 1) candidate entity selection in a knowledge base that may refer to a given entity mention in the text; 2) similarity score definition for each selected candidate entity. Approaches to candidate entity generation are mainly based on string comparison between the textual representation of the entity mention in the text and the textual representation of the entity in the knowledge base. A wide variety of techniques make use of redirect pages, disambiguation links, and hyperlinks in the text to build a Name Dictionary that contains information about the named entities and provides a good base for linkage possibilities, as in [24], [25], and [26].

Surface form expansion helps to find other variants for the surface form of the entity mention, for example, abbreviations that are extracted from the context of the processed document as in [27], [28], [29], and [30]. Although some candidate generation and ranking features demonstrate robust and high performance in some data sets, they could perform poorly in others. Therefore, when designing the features of the entity linking systems, a decision must be made regarding many aspects, such as the trade-off between accuracy and efficiency, and the characteristics of the applied data set [23].

Using name dictionary-based techniques is not usable in domains such as urban planning and development, since the terms in the domain-specific ontology are similar and some of them share common words, for instance, “lokalita” (en. “locality”), “zastavitelná lokalita” (en. “buildable site”), and “zastavitelná stavební lokalita” (en. “buildable construction site”). Therefore, using features such as entity pages, redirect pages, hyperlinks, and disambiguation pages as in [24], [25], and [26], bag of words [31] and entity popularity [32], are not useful in our case. Even statistical methods give poor results due to the small corpus and lack of training data.

Only a few attempts have been made to tackle this area of research for the Czech language. For example, the authors in [33] created a Czech corpus for a simplified entity linking task that focuses on extracting instances of the class

"Person". Building such a corpus is a costly task considering the different types of domain-specific entities that exist in our data.

The next task is to calculate a proper score for each candidate entity. In [34] and [35], the researchers used a binary classifier to tackle the problem of ranking candidate entities. This method needs many labeled pairs to learn the classifier, and it is not a final-decision method since the final result set can contain more than one positive class for an entity mention. While researchers in [36] and [37] treated the entity ranking problem as an information retrieval task, probabilistic models are also used to link entity mentions in web free text with a knowledge base. The work in [38] proposed a generative probabilistic model that incorporates popularity, name, and context knowledge into the ranking model.

In recent years, researchers have paid increasing attention to automatically analyzing textual safety reports in different domains. In the transportation domain, such as analyzing maritime accident investigation reports [39] and railroad accident investigation reports [40] and traffic accidents reports in [41]. Safety reports were also studied in other domains, such as the construction safety domain [42] and [43], and in the medical safety domain, [44].

2.2.2 Ontology Learning From Text

Ontology learning and population methods can be divided into clustering-based approaches that make use of widely known clustering and statistical methods, and pattern-based approaches that mainly employ linguistic patterns. However, the former approaches require large corpora to work well. Additionally, English is the only language supported by most of the available ontology learning systems [2].

Several linguistic-based techniques are used to perform almost all the tasks in ontology learning. Seed words provide a good starting point for the discovery of additional terms relevant to that particular domain [45], [46]. Syntactic structure analysis and dependency analysis examine syntactic and dependency information to uncover terms and relations at the sentence level [47]. The use of lexico-syntactic patterns was proposed by Hearst [48] and has been employed to extract hypernyms [47].

Several clustering-based techniques are applicable to tasks of ontology learning. In [49], clustering with a measure of similarity is used to assign terms into groups for discovering concepts or constructing hierarchy. Conditional probabilities of the occurrence of terms in documents are employed to discover hierarchical relations (subsumption) between them [50]. Also, association rule mining is employed to describe the associations between concepts at the appropriate level of abstraction [51].

In [52] researchers focus on presenting a method for learning axioms from text based on Named Entity Recognition. [53] describes a new approach to ontology learning that consists of a method for the acquisition of concepts and their corresponding taxonomic relations, where axioms such as *disjointWith* and *equivalent* classes are learned from text without human intervention. [53] focuses on identifying the relationships between medical concepts as defined by the REMed (Relation Extraction from Medical documents) solution that aims to find the patterns that lead to the classification of concept pairs into concept-to-concept relations.

Two types of patterns can be applied to natural language corpora. Lexico-syntactic patterns use lexical representations and syntactical information, and lexico-semantic patterns combine lexical representations with syntactic and semantic information in the extraction process. Text2Onto [54] combines machine learning approaches with basic linguistic processing to extract relations from text. FRED [55] is a tool for automatically producing RDF/OWL ontologies and linked data from natural language sentences. Neither of the tools provides direct support for documents in the Czech language. Java Annotation Patterns Engine (JAPE) [56] is a language for expressing patterns within the open-source platform General Architecture for Text Engineering (GATE) [57]. Researchers define patterns using JAPE rules, taking advantage of the linguistic preprocessing components provided by the GATE framework as in [58]. However, GATE does not have models to support resources in the Czech language. Much cleaner rules with considerably less effort and time to create can be written using Hermes Information Extraction Language (HIEL) [59].

In [58], researchers defined a set of lexico-syntactic patterns corresponding to Ontology Design Patterns (ODPs), namely *subClassOf*, *equivalence*, and *property* rules. Lexico-semantic patterns were defined focusing on domain-specific event relation extraction from financial events in [60], and in [61] to spot customer intentions in micro-blogging. To the best of our knowledge, no work has been done on the topic of lexico-semantic patterns for low-resource languages such as Slavic languages. In this work, we attempt to define a preliminary set of these patterns corresponding to *subClassOf*, *equivalence*, *part-whole*, and *hasProperty* relations.

2.2.3 Tools

There are many tools that provide support for building an ontology from unstructured text. A list of these tools can be found in Figure 2.1, as presented in [55]. The list shows the different tasks that the tools provide in the ontology learning process. Most of these tools focus mainly on the NER task, which is not useful in the case of building a domain ontology. Accurate Online Disam-

biguation of Named Entities in Text and Tables (AIDA)[3] is a framework and online tool for entity detection and disambiguation. It provides mappings between mentions in a given natural language text based on specific types from the YAGO2 knowledge base[4]. The supported types are person, artifact (e.g. movie, publication, system, etc.), event (e.g. battle, election, etc.), organization, and yagoGeoEntities. This is a limitation in the case of extracting domain knowledge where, when tested with domain-specific data, it almost did not retrieve any useful information. A similar experience was gained with tools such as **DBpedia Spotlight**[5], where the acquired data were limited to generic entities existing in the DBpedia ontology[6]. For example, in the following sentence taken from an aviation-related document, "An aluminum fuel tank is installed in each of the wings", DBpedia annotated the word "tank" with the entity "dbpedia[7]:Tank" referring to an *armored fighting vehicle*, which is, obviously, not the case.

Tool	TopE	NER	NEReS	TE	TReS	Senses	Tax	RE	Events	Roles and Frames
AIDA[a]	–	.89	**.80**	–	–	.64	–	–	–	–
Alchemy[b]	**.52**	.89	–	.20	–	.64	–	.30	–	–
Apache Stanbol[c]	–	.77	.25	–	–	.50	–	–	–	–
CiceroLite[d]	–	.89	.75	.21	.07	.64	–	.25	.18	.22
DB Spotlight[e]	–	.79	.55	–	–	.42	–	–	–	–
FOX[f]	–	.86	.75	.33	**.65**	.57	–	–	–	–
FRED	–	.84	.60	**.90**	.07	.48	**+**	**.82**	**.87**	**.69**
NERD[g]	–	.88	.60	–	–	.69	–	–	–	–
Open Calais[h]	.48	.82	–	–	–	.57	–	–	.04	–
PoolParty KD[i]	.28	–	–	–	–	–	–	–	–	–
ReVerb[j]	–	–	–	–	–	–	–	.27	–	–
Semiosearch[k]	–	–	.60	–	.46	–	–	–	–	–
Wikimeta[l]	–	.86	.75	.04	.07	**.80**	–	–	–	–
Zemanta[m]	–	**.93**	–	–	–	.27	–	–	–	–

Figure 2.1: Summary of evaluation results for basic ontology learning tasks performed by different tools [55]

On the other hand, **Apache Stanbol**[8] is an open-source HTTP service that provides the ability to work with custom vocabularies and create custom indexes upon them. It also comes with a list of enhancement engine implementations, with the ability to build a specific one to get the most benefit out of the tool. However, this can fit only to the entity linking task, since to make use of the tool, it is necessary to feed the engine with a domain ontology as input. Also, there is no possibility of building custom extraction rules to enrich the ontology.

For the task of term extraction, **FRED** [55] achieves the highest possible accuracy with 90%. In addition to Named Entity Recognition (using Apache Stanbol and TagMe[9]), it provides coreference resolution using CoreNLP[10] and word sense disambiguation. The functionality can be tested using the pub-

[3]http://www.mpi-inf.mpg.de/yago-naga/aida/
[4]http://www.mpi-inf.mpg.de/yago-naga/yago
[5]https://www.dbpedia-spotlight.org/demo/
[6]https://www.dbpedia.org/
[7]dbpedia: http://dbpedia.org/page/
[8]https://stanbol.apache.org/
[9]https://tagme.d4science.org/tagme/
[10]https://stanfordnlp.github.io/CoreNLP/

licly available REST service[11] with a restricted number of requests (1500 per day). For the input text, the FRED service outputs an RDF document pointing to mentions within the text. The RDF provides type and taxonomy induction, temporal relation extraction from tense expressions, adjective semantics, modality, and negation representation, etc. Furthermore, the output is expressed using DOLCE+DnS Ultralite ontology[12], which is an upper-level ontology based on principles similar to our domain ontology that we built based on the UFO (Unified Foundational Ontology) upper-level ontology. On the other hand, FRED, like many other tools in this list, supports only natural language text written in English. It does not provide support for other languages, such as Czech, Chinese, etc. Older versions of FRED used the BING API[13] to translate the actual text into English prior to processing, which did not provide a sufficient quality output. This feature is not available in the latest version of the tool.

GATE is a full-lifecycle open-source solution for text processing that provides several language resources as well as processing resource components to perform various NLP tasks. It allows easy reuse and combining of basic NLP modules. It provides tools and plugins that support working with ontology, creating and managing semantic annotations, and allows the use of semantically annotated documents to add new facts to the knowledge base. Gazetteers within the GATE framework refer to lists of (typically named) entities, for example, the list of all countries or months of a year. They are essential components to perform the NER task. It is possible to build such gazetteers based on ontology as input, and GATE provides the possibility of this transformation using the OntoRoot Gazetteer and the Flexible Gazetteer.

OntoRoot Gazetteer This gazetteer can be found as a plugin within GATE. It can be created dynamically by providing the desired ontology as input and is capable of producing ontology-based annotations when combined with other GATE processing resources.

Flexible Gazetteer This gazetteer provides the flexibility to perform the lookup over a document based on the values of an arbitrary feature of any annotation type. This gives the possibility to perform the matching on the root level (i.e., lemma) of tokens in the text (the output of the morphological analysis, for example). It allows the usage of any other type of gazetteer, for our purpose, custom OntoRoot Gazetteer. The flexible gazetteer can be found in the list of plugins provided by GATE.

Hermes News Portal (HNP) implements the lexico-semantic **Hermes Information Extraction Language (HIEL)**. HIEL is an expressive language

[11] http://wit.istc.cnr.it/stlab-tools/fred/swagger.json
[12] http://ontologydesignpatterns.org/wiki/Ontology:DOLCE+DnS_Ultralite
[13] https://azure.microsoft.com/en-us/services/cognitive-services/bing-web-search-api/

for specifying Lexico-Semantic Patterns that makes use of regular expressions over ontology concepts. HIEL is more expressive and less verbose than JAPE rules within the GATE framework [59]. HNP implements HIEL Information Extraction Engine (HIEE) plugin that consists of two main parts, the preprocessing stage which heavily relies on GATE components, and the rule engine that compiles the rules in the rule compiler and matches these rules with the text using the rule matcher.

2.2.4 Temporal Information Extraction

Many temporal expression extractors have been proposed in the literature. Temporal information extraction systems usually use rule-based methods, sequence segmentation machine learning algorithms such as hidden Markov models, or Conditional Random Fields as in [62], or hybrid methods that combine both techniques, as in [63] and [64]. HeidelTime [65], an open source system for temporal expression extraction, is a representative rule-based system that performed well in TempEval2[14] competition. The Stanford temporal tagger (SUTime) [66] is one of the best temporal taggers currently available with a 90.32 F measure score in TempEval-3 with the English Platinum Test set [67]. SUTime annotations are automatically provided with the StanfordCoreNLP[15] pipeline by including the Named Entity Recognition annotator. SUTime is a rule-based extractor, which means that it can normally be configured to use rules specified in external files that are convenient for the data being analyzed.

The Semantic Web provides a suitable environment for representing temporal data. Web Ontology Language (OWL)[16] through the annotation properties of standard vocabularies, for example, DCMI Metadata Terms[17] gives the ability to incorporate time entities into existing ontologies by representing temporal knowledge and time-based information. Many ontologies were proposed to represent temporal information in a structured way. The OWL-Time ontology [12] is an ontology of temporal concepts, to describe the temporal properties of resources. The OWL-Time ontology is further discussed in Section 2.1.1.3

[14] http://www.timeml.org/tempeval2/
[15] nlp.stanford.edu/software/corenlp.shtml
[16] https://www.w3.org/TR/owl-ref/
[17] http://purl.org/dc/terms/

Chapter 3

Methodology

One goal of this work is to provide an end-to-end methodology for information extraction and ontology learning from unstructured documents to continuously develop the domain ontology that covers a specific domain of interest. We mainly focus on building domain ontology based on limited input resources in the studied domain, where the limited resources may refer to a small set of input documents, little or no annotated data, or textual documents written in a low-resource language that lacks the support of tools or processing methods. The challenge of working with such languages makes the development of methods for building domain ontology even more critical. A domain ontology is a shared explicit representation of concepts and relationships within a specific field or area of knowledge that can be further utilized by domain applications and various users. To build the domain ontology, this work uses two main inputs: a seed ontology O in OWL format with an initial set of ontology classes, and the set of relations between these classes that can be built in the early stages of the ontology development, for example, by domain experts, or even crawled from public or internal sources, and a non-empty set of related documents T in the studied domain.

The seed ontology is the ontology in its early development stage, and can be as simple as a set of main keywords in the studied domain, and provides a starting point for the construction of the domain ontology. It serves as a foundation that can be built upon in an iterative manner, with each iteration resulting in an evolved ontology based on the domain-related documents that represent a source of additional information about the domain and are used to augment the seed ontology.

The process of building the domain ontology is done in multiple phases. The phases are *1) the Knowledge Acquisition phase* where the seed ontology is first acquired, *2) the Entity Linking phase* where we use the created preliminary ontology to create links between the entities in the knowledge base and the mentions of these entities in the text, and *3) the Ontology Learning phase*

where we apply Lexico-Semantic Patterns (LSPs) to the annotated text and extract new information to enrich the ontology automatically with more entities and relations. The *Entity Linking* and *Ontology Learning* phases are the core steps and can be performed in an iterative fashion. Acquiring the seed ontology in *Phase 1* can be done in multiple ways (manual, semi-automatic, or automatic) and it is out of the scope of this study. In the context of the following experiments, we use available ontologies as seed ontology when possible. In addition, we utilized a simple statistical keyword extractor to spot some of the key concepts that can be implemented as a seed ontology after being reviewed by subject matter experts. Finally, domain experts can be involved in any of the phases to control the quality of the resulting ontology. The approach steps are depicted in Figure 3.1.

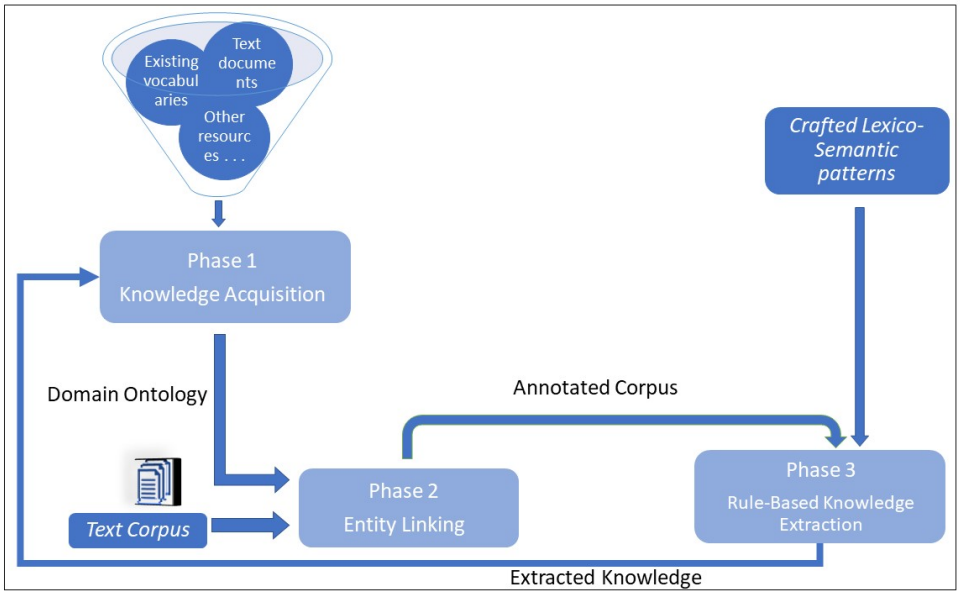


Figure 3.1: Entity Linking and Ontology Learning Methodology

In the following, we discuss each phase in detail.

3.1 Knowledge Acquisition

Knowledge Acquisition for Ontology involves activities to capture knowledge (e.g. concepts, instance data) from diverse sources, such as documents, experts, databases, and others. We use this general term to label *Phase 1* as it is a phase where any of these activities and their combinations might be useful. Although activities in *Phase 2* and *Phase 3* could also be considered knowledge acquisition activities, we pulled them out into separate phases in order to emphasize the importance of those activities with respect to our document corpus and use cases.

When considering the first iteration of the process described in Figure 3.1, we need to build our initial domain ontology, which would be used for the next

phases and iterations of the described process. Acquiring the seed ontology is an initial optional step and depends solely on the available resources in the studied domain and might differ between various use cases. It can be done manually (e.g. using manual annotations of textual resources), or semi-automatically using a combination of structured data sources that would be further analyzed and refined by the domain experts. A preliminary analysis of the textual documents can also help reveal the main classes in the domain using, for example, the famous Term Frequency - Inverse Document Frequency (TF-IDF) approach widely used in information retrieval. The seed ontology can then be formally constructed on the basis of the preliminary set of the identified entities.

Within later iterations, the main goal of this phase is to merge the extracted knowledge from other phases through the validation of the knowledge by automated tests or by domain experts. The phase output is a validated domain ontology that is ready to serve as input for *Phase 2*, the entity linking step.

3.2 Entity Linking

In this step, the task is to identify the mentions in the text with their corresponding ontological entities. When working with textual documents, it is necessary to perform a natural language processing step to enhance the parts of the text with additional syntactic, morphological, and semantic information. For example, tokenization, sentence splitting, and POS tagging are part of the pre-processing pipeline that provides a proper input for the information extraction and entity linking task, which, in turn, provides the semantic context.

Once the text is pre-processed, the entity linking task can be performed, which involves identifying the mentions in the text that correspond to ontological entities. This is typically done by comparing the text against a knowledge base to determine which entities are mentioned in the text and to disambiguate between potentially ambiguous mentions.

We identify two main problems when performing entity linking for a limited-resource domain. The first is when the corpus size is limited. For example, in an aviation safety reporting system, the corpus is built gradually and reports are mainly required to be analyzed individually once the report is submitted to the system. Another issue is the lack of annotated data in the studied domain. For these cases, we combine different entity linking systems and techniques outputs in order to generate the most accurate results. This case is explained in further detail in Sections 4.1 and 4.2.

The second problem is when the corpus is comprised of documents written in a low-resource language where there are no tools available that can perform the entity linking task. In this case, we introduce a vanilla approach to entity

linking that involves providing links back to the entities that are mentioned in the text. The approach is applied to the Czech language resources as a proof of concept but can be extended to a wide variety of languages. The extraction pipeline consists of a preprocessing step that performs the necessary NLP tasks. For example, splitting the text into tokens, stop-word removal, and providing the lemmas for each token. A suitable morphological analyzer can be used that can handle the specific language. Similar pre-processing should be performed on the ontological classes in the ontology. The next step is to generate the set of candidate entities. Meaning, for each token in the text, we find candidate entities in the corresponding available labels of the ontological terms. Finally, a scoring function is applied to each candidate to choose the best match. This approach is explained in greater detail in Section 4.1.

The end result of this process is contextualized documents T' with a set of annotations that link each mention in the text to a corresponding ontological entity from the seed ontology O, providing a way to understand the meaning of the text in a structured way. These annotations can then be used as input to downstream tasks, such as question answering, information extraction, or, in this case, enhancing the knowledge base with new semantic entities, which is explained in the Ontology Learning section.

3.3 Ontology Learning

Even though the domain ontology that we acquired in *Phase 1* (the seed ontology) is rich, it is still far from complete as the knowledge evolves. The output of the Entity Linking task in *Phase 2*, the contextualized semantically augmented text T' is considered input to this phase, besides the ontology O. We also take advantage of the morphological analysis performed in the entity linking step to provide syntactic information about the text being processed. By utilizing the combination of syntactic and semantic information, we create a set of Lexico-Semantic Patterns (LSPs) to build the ontology taking into consideration the characteristics of the special corpus and the language used.

These rules may be written in a variety of formats. They are generally designed to capture a particular syntactic or semantic structure in the text, such as a specific word sequence, a particular dependency relationship, or a co-occurrence pattern between entities. This work focuses on extracting relationships between entities, mainly the *part-whole*, *subClassOf*, *equivalence*, and *hasProperty* relations. This approach is discussed in greater detail in Sections 5.1 and 5.2.

Once the rules have been applied to the text, the output can then be structured and augmented back to the ontology O' where additional concepts and

relations are extracted from applying LSPs to T' and O. This structured output can then be used for a variety of downstream applications, such as knowledge management, decision support, or, in this case, knowledge-based information extraction.

Other types of information can also be extracted from the text, such as temporal and spatial knowledge, which can be useful to enrich the ontology. We provide an enhancement to the ontology classes by providing the temporal dimension as discussed in 5.3.

3.4 Continuous Ontology Learning

We consider the Entity Linking performed in *Phase 2* and the Ontology Learning in *Phase 3* as non-separate tasks where domain ontology provides background knowledge supporting identifying and linking entities in the text. Then, the linked terms in the textual documents can help to reveal potential classes and relations to extend the ontology, which, in turn, helps to detect more information from the corpus. In other words, more extracted information leads to a richer ontology and a richer ontology leads to extracting more information.

In order to better understand the iterative system involving *Phases 2 and 3*, consider the following scenario:

1. A seed ontology O in OWL format with an initial set of ontology classes $C = \{C_1, C_2, ..., C_n\}$ and the set of relations between these classes $R = \{R_1, R_2, ..., R_m\}$ that can be built in the early stages of ontology development as in *Phase 1*, and a non-empty set of related documents T in the studied domain, as input to *Phase 2* - the entity linking step of the continuous ontology learning system.

2. The Entity Linking phase processes the input as described in *Phase 2* and produces contextualized documents corpus T' where each entity mentioned in the text is identified with the corresponding ontological entity from the seed ontology O.

3. The annotated corpus T' produced by the entity linking system is then used as input to the ontology learning in *Phase 3*, together with the domain ontology being developed O and a set of Lexico-Semantic Patterns (LSPs). The ontology learning pipeline applies the set of LSPs to the annotated text and identifies the corresponding relations between the concepts in the ontology O and the additional concepts to be added in the ontology hierarchy.

4. The output of the previous step can then be structured and augmented back to the ontology O', where $C' = C \cup \{C'_1, C'_2, ...C'_n\}$ and $R' = R \cup$

$\{R'_1, R'_2, ..., R'_m\}$, where C' and R' were extracted from applying LSPs to T' and O.

In the next iteration, the additional knowledge in the ontology can be used to extract further information from the evolving textual corpus. By linking the entities in the corpus using the ontology O', the resulting textual corpus T'' is then further analyzed using the LSPs which in turn reveal further concepts and relations producing an evolved version of the ontology denoted as O''.

The ontology learning process is initiated by updating the ontology or updating the corpus with new documents.

During the entity linking phase, the extracted text entities in the documents are semantically mapped to existing ontological entities (concepts and relations).

The iterative text annotating and ontology learning methodology potentially guarantees a more reliable understanding of the new unseen documents that need to be processed, and in turn enhances the quality of the annotated corpus, as well as enriching the domain ontology with new terms and relations, and supporting the complicated and expensive process of building the domain ontology.

To further illustrate the benefit of iterative ontology learning, consider the following example from the aviation safety and reliability domain.

1. Seed ontology with the defined classes *Aircraft*, *Wing*, *Engine*, and *Fuselage*.

2. Consider the input text containing the following text, *"The wing is an integral part of an aircraft that provides lift during flight."* The entity linking phase result will be "wing" linked to the "Wing" class, and "aircraft" linked to the "Aircraft" class.

3. In the learning phase, and with appropriate LSPs designed to extract the partonomy relationship, the system extracts the *partOf* relation between *Wing* and *Aircraft*. Moreover, from the sentence "The landing gear, as part of the aircraft, facilitates safe takeoffs and landings", the LSPs identifies the concept *"Landing gear"* and adds it to the ontology.

4. In a further iteration, for the sentence, "The landing gear can be categorized into retractable landing gear and fixed landing gear." The EL system will be able to link the mention of "landing gear" to its concept in the ontology. The *"Landing gear"* concept was added to the ontology as part of the learning phase in the previous step. Applying the *subClassOf*

relationship LSPs to the annotated sentence will reveal two subclasses to be added to the ontology, *Retractable landing gear* and *Fixed landing gear*.

As seen in the example, each step reveals additional information to be added to the ontology, resulting in the ontology evolving with each iteration. The concepts and relationships learned are affected by the design of the LSPs used, and can be fine-tuned depending on the individual studied use case.

3.5 Ontology Quality Assurance

Fully automatic learning of ontologies might not be possible [15]. Fully automation of such a system is hard to achieve due to the ontology learning evaluation task. Creating ontologies automatically does not guarantee the quality of these ontologies. We want the ontology that we create to keep a reliable level of quality.

Quality assurance in building ontology based on text is generally the process of ensuring the validity, reliability, and accuracy of the created ontology. It aims to produce a reliable and high-quality ontology that effectively represents the knowledge extracted from textual sources. It involves a thorough evaluation, error detection, and validation processes to ensure the usefulness and applicability of the ontology in the intended domain.

In order to achieve that, various techniques and methodologies can be applied to evaluate the quality of the created ontology and address any issues or errors that may arise during the learning process.

Supervised attempts are certainly difficult to apply due to the bottleneck introduced by the interaction of a domain expert and the great effort required to compile a large and representative evaluation set. Experts in the domain possess deep knowledge and understanding of the subject matter and their input can greatly contribute to the reliability and relevance of the learned ontology. However, reviewing all the extracted entities by experts might be tedious work, impractical, and not user-friendly. There are many ways in which this can be improved. One example is by generating a set of competency questions, which are specific questions that are designed to evaluate the ontology's ability to answer important queries in the corresponding domain. By formulating these questions, it is possible to assess whether the learned ontology captures the necessary knowledge and exhibits the desired behavior.

Moreover, each task in the learning pipeline can be evaluated separately to ensure the best achievable results that can be used as input to the next step. For example, the output of the entity linking system can be evaluated prior to being used as input to the ontology learning step, and similarly, the output of

applying the LSPs on the annotated text can be scored and evaluated before augmenting the resulting ontology.

Choosing the method for evaluating the ontology learning technique can vary based on each use case as there is no generic framework that can fit all situations. A combination of these methods or tailored approaches can be employed to ensure the highest possible quality and accuracy of the learned ontology. For that reason, the specifications of the ontology quality assurance methods are beyond the scope of this study. However, in the framework presented, we used common methods to evaluate each step of the learning pipeline. For example, to evaluate the entity linking system, common evaluation metrics such as precision, recall, and the F1 measure were used. Additionally, each annotated mention of an entity is attached with an accuracy score that is further reviewed by an expert to verify the linking accuracy. Similarly, the output of the LSPs is revised by the domain experts who decide whether the extracted entity fits and refine the final result. Furthermore, the ontology is then checked by domain experts for any discrepancies or contradictions, identifying and addressing errors or inconsistencies and resolving them to maintain the integrity of the learned ontology.

Chapter 4

Entity Linking

The first step to enhancing the ontology is to match the existing ontological entities with their mentions in the textual documents.

In this section, we combine multiple information extraction tools and techniques in the processing pipeline to extract information and provide links back to the ontology. We describe our efforts to extract information from an unstructured textual corpus based on a seed ontology in the respective domain with low resources. We identify two issues that we discuss in this section. First, when available documents are written in a low-resource language that is undersupported to perform the entity linking task, and when the training data in the domain of interest are limited.

4.1 Limited-Resources Language

According to [68], most of today's NLP research focuses on 20 of the 7000 languages of the world, leaving the vast majority of languages understudied. Limited-Resources Language (LRL) are languages that lack common resources that are essential to perform entity linking as well as ontology learning tasks, such as dictionaries, ontologies, NER, or linking tools. This can make it more difficult to develop and evaluate machine learning models for these languages and to ensure the accuracy of the resulting ontology.

4.1.1 Urban Planning and Development - Corpus

The corpus in this task is a set of documents and vocabularies related to these documents in the domain of urban planning and development. The documents are on different levels of detail regulating spatial and urban planning in Prague. All documents are in Czech. The main document in this set is the Metropolitan Plan of Prague (MPP)[1] which is a spatial plan for the Czech capital. It consists

[1] `https://plan.iprpraha.cz/uploads/assets/prohlizeni/zavazna-cast/textova-cast/TZ_00_Textova_cast_Metropolitniho_planu.pdf` accessed: 2023-05-29

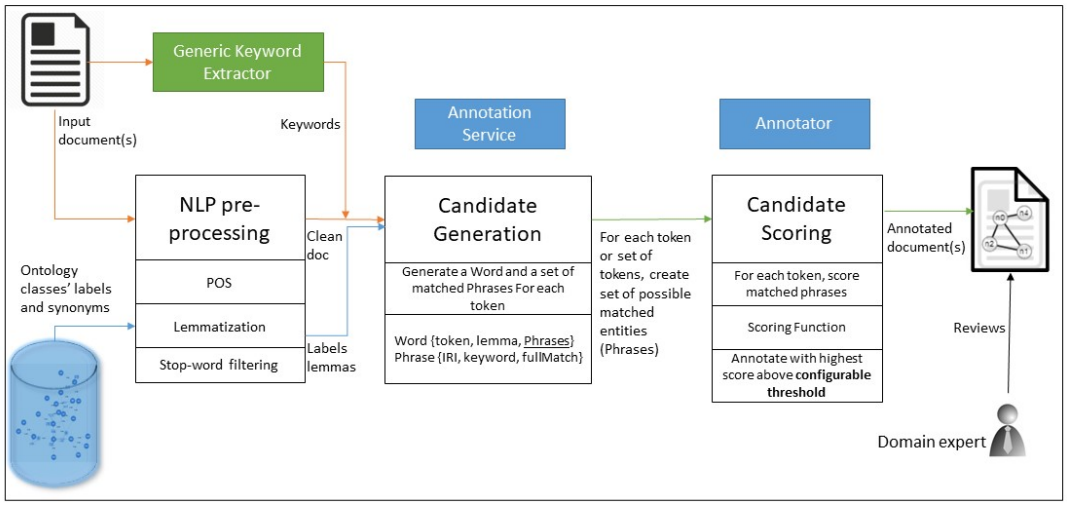


Figure 4.1: Entity Linking Pipeline

of 168 articles divided into ten parts. The current version of the MPP vocabulary corresponding to this document contains 59 terms. Other documents including but not limited to the document of the *Law 2006/183 Col., Building Law*[2], the law of urban planning and building regulations in the Czech Republic and the *Prague Building Regulations*[3] in a version of 2016 *(PSP 2016)*. The *Building Law* has 179 paragraphs divided into seven parts, and its corresponding vocabulary has a few tens of terms currently. On the other hand, *PSP 2016* which regulates the construction of buildings and urban planning in the Czech capital, is conceptualized as a book with 202 pages, describing 87 paragraphs, and the PSP2016 vocabulary consists of 102 terms.

4.1.2 Entity Linking Pipeline

Since the corpus is in the Czech language, which, like many other low-resource languages, lacks any existing tools that solve the entity linking and extraction task, in this section we introduce a vanilla approach to provide links between ontological entities and their mentions in the textual documents. The approach can be summarized in a *preprocessing* step, the *candidate entity set generation* **(Annotation Service)** step, and the *candidate entity set scoring* step **(Annotator)**, as shown in Figure 4.1.

Algorithm 1 describes the Entity Linking approach. In the following, we describe in greater detail each of the steps in the processing pipeline.

[2] `https://www.zakonyprolidi.cz/cs/2006-183` accessed 2023-05-29
[3] Not available online

Algorithm 1 Entity Linking

```
1: procedure EntityLinking(ontology, text)
2:     lemmatizedOntology ← lemmatizeOntologyLabels(ontology)
3:     processedText ← processInputText(text)
4:     annotationResults ← {}
5:     for each token in processedText do
6:         matchedPhrases ← getMatchedPhrases(token)
7:         if matchedPhrases ≠ empty then
8:             word ← createWord(token, matchedPhrases)
9:             annotationResults.append(word)
10:        end if
11:    end for
12:    for each word in annotationResults do
13:        if word.phrases ≠ empty then
14:            sequence ← findLongestMatchedSequence(word)
15:            if sequence ≠ empty then
16:                bestPhrase ← calculateBestPhrase(sequence)
17:                if bestPhrase.score > threshold then
18:                    annotateToken(word.token, bestPhrase)
19:                end if
20:            end if
21:        end if
22:    end for
23:    return annotationResults
24: end procedure
```

4.1.2.1 Preprocessing

Any task dealing with textual documents needs to perform a Natural Language Processing step to enhance the parts of the text with further syntactic pragmatic, and morphological information. Some of the steps performed include tokenization, sentence splitting, and POS tagging, which are handled by a morphological analyzer tool appropriate to the language being dealt with.

For the Entity Linking task, morphological analysis is important because Czech, like many other Slavic languages, is a highly inflective language. Meaning that a word can have different suffixes to determine a linguistic case so that tokens can have many forms belonging to the same lemma and referring to the same semantic entity. For example, **"Metropolitní plán"** (en. "Metropolitan plan") can appear in several forms like "*Metropolitním plánem*", "*Metropolitního plánu*" and so on. We perform the same processing on the labels of entities in the ontology for the same reason.

The next step is the removal of stop-words to reduce the number of tokens to be matched in the document. A list of such stop words for the Czech language is defined in [69]. We extend this list to include more unnecessary words derived from our data. The full version of this list can be found in Annotace Github

repository[4].

After stop-words removal, it is necessary to match all the remaining tokens since, in the text, most of the tokens might refer to a semantic entity in the ontology. Using regular Named Entity Recognition tools would not be enough to recognize all potential mentions. This is because the ontological classes are diverse and not necessarily limited to the standard named entity classes such as geographic location, person, or organization.

4.1.2.2 Candidate Entity Set Generation

At this point, we have the clean document enriched with lemmas that should be linked to the corresponding semantic classes. First, we find candidate entities in the ontology that may refer to tokens in the text. We apply the famous Jaccard similarity coefficient algorithm to lemmatized tokens taking into consideration the lexical matching. i.e., the string comparison between the surface form of the entity mention and the name of the entity existing in the knowledge base.

As mentioned earlier, our method is based mainly on three aspects: the string similarity measures of the tokens and the candidate entity name, the number of matched tokens, and the order of these tokens as they appear in the text to ensure a final decision result.

Given a vocabulary V that has a set of entities E and a processed document D composed of a set of potential entity mentions M_d, we need to find for each entity mention $m \in M_d$ (in our case a sequence of tokens) a mapping to its corresponding entity $e \in E$. In many cases, it can happen that the mapping is not injective, since there are more candidate entities in the vocabulary to be linked to a specific mention. Thus, it is necessary to rank the entities in the candidate set to choose the most relevant entity and associate it with the sequence of tokens that is considered to be an entity mention of the semantic entity.

For every single token (one word), the *Annotation Service* retrieves all possible entities to which the surface form of this token could refer and creates a set of candidate entities for this token E_t. We refer to these annotations as *Phrases*. A *Word* contains information such as the surface form of the single token against which we match the entities, the lemma, and a list of matching phrases. A *Phrase* contains information like the URI of the retrieved entity in the ontology, whether the token is a "fullMatch" to the entity label, and if the token is recognized as an important keyword by a generic keyword extractor.

[4]https://github.com/kbss-cvut/annotace

4.1.2.3 Candidate Entity Set Scoring

Even if a phrase indicates a "fullMatch" to the token, it does not mean that this token will be annotated with this phrase. The *Annotator* takes into account the neighbors of this token while deciding on the annotation. That means that it looks around the token and gives a higher score to the phrase if the label of the entity has common substrings with the tokens around. In other words, if in text M_d occurs the sequence $t_1t_2t_3$, t_1 that matches the label of the entity e_i in the ontology, but the sequence of tokens, t_1t_2 matches another entity e_j in the ontology, then the service will give a higher score to annotate the multi-word mention t_1t_2 with the entity e_j. In case there is an entity e_k in the ontology with a label that also matches the third token, the sequence $t_1t_2t_3$ will be annotated with the entity e_k. For example, assume that the document contains the sequence of tokens **"součást otevřené krajiny"** (en. "part of an open landscape"), and in the vocabulary there is e_1 : <mpp:otevřená-krajina>, e_2 : <mpp:krajina>, the mention **"otevřené krajiny"** will be annotated with the entity e_1.

To test the entity linking approach presented, **Annotace**[5], a text annotation service is implemented and used in the context of TermIt[6], a terminology management tool based on Semantic Web technologies developed at Czech Technical University in Prague. We discuss the implementation in greater detail in Section 6.1.

4.1.2.4 Results and Evaluation

To evaluate the entity linking system, we used the set of documents and vocabularies described in Section 4.1.1. Text files are loaded into TermIt and automatically annotated using the vocabulary related to the respective documents. The annotations are then revised by a human expert and evaluated based on the precision, recall, and F1 measures. The scores are calculated as follows: the True Positives (TP), the number of correct links suggested by Annotace, the False Positives (FP), where the links are suggested by Annotace but are false, and the False Negatives (FN), the number of mentions in the text that are not suggested by Annotace as a term occurrence but the term is present in the vocabulary. These statistics are then used to calculate the well-known precision, recall, and F1 measures.

$$Precision = \frac{|\{correctly\ linked\ entity\ mentions\}|}{|\{links\ generated\ by\ Anotace\}|} = \frac{tp}{tp + fp}$$

[5]Source code is available at `https://github.com/kbss-cvut/annotace` accessed: 2023-05-29

[6]`https://github.com/kbss-cvut/termit` accessed: 2023-05-29

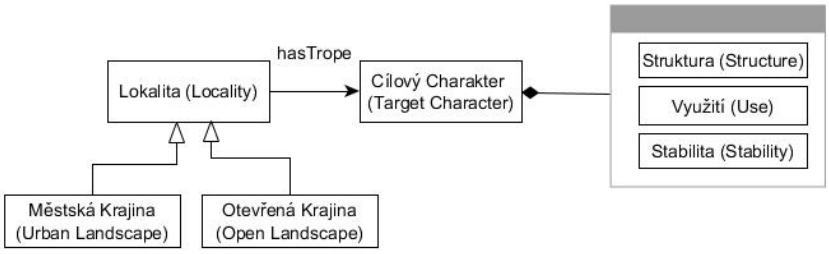


Figure 4.2: Example of involving the hierarchy of the ontology in the disambiguation task

$$Recall = \frac{|\{correctly\ linked\ entity\ mentions\}|}{|\{entity\ mentions\ that\ shouldd\ be\ linked\}|} = \frac{tp}{tp + fn}$$

$$F_1 = \frac{2.Precision.Recall}{Precision + Recall}$$

Annotace achieved average precision, recall, and F1 measures of 83%, 79%, and 80.9%, respectively. It is noticeable that false negatives occur more often than false positives. There are only a few distinct false positives. In most cases, terms are defined in the vocabulary and used with different meanings in the context of the document. As illustrated in Figure 4.2, in the vocabulary, it happens that the term **"Lokalita"** (en. "Locality") has intrinsic trope **"Cílový charakter lokality"** (en. "Target character of locality") which, in turn, is composed of other intrinsic tropes like **"Struktura"** (en. "Structure"), **"Stabilita"** (en. "Stability"), and **"Využití"** (en. "Usage") and in most of the false positive cases, the word "Struktura" is used in a different context. For example, in the following sentence, *"Metropolitní plán je především plánem struktury území"* (en. "The metropolitan plan is primarily a plan of the area structure"), the word "Struktura" is recognized as the term **"Struktura"** in the vocabulary even though, in this sentence, it means the structure of the area (in Czech, "Území") and is not meant to describe the structure of the locality. The link in this case should not be suggested, and hence, it is considered a false positive. To solve this problem, the specialization classes of the class **"Lokalita"** should be considered in the disambiguation process, which we will consider in future work.

On the other hand, false negatives occurred while evaluating the MPP document when some frequently used terms come from other vocabularies and are not present in the vocabulary of MPP and hence, Annotace is not able to retrieve those terms correctly without involving other vocabularies in the process. However, it is possible to include a list of input vocabularies in the request, which can potentially improve the recall.

It is still noticeable that most of the false negative cases occurred due to lemma mismatching between the surface form and the term in the ontology when the morphological tagger erroneously returns different lemmas for the same string. This means that the quality of the lemmas provided by the morphological analyzer plays a big role in the accuracy of the linking system.

The approach can be easily adapted to provide entity linking for a wide variety of other languages. Most parts of the implementation of the text analysis service for the Czech language **Annotace** can be reused. The only requirement is to plug in the desired morphological analyzer that is capable of normalizing the text into tokens and their lemmas, which is the only step that requires language-specific processing. Optionally, a list of stopwords can be provided for the language and domain studied.

4.2 Limited Training Data

Domain-specific ontology-based information systems can benefit from identifying entities and relationships in textual assets in the system. However, for many domains, the textual corpus can be either small or is being built gradually by the system users. Moreover, there is a lack of quality annotated documents in the domain of interest. These factors can limit the effectiveness of building sufficient models specific to this task, as building such models requires a vast amount of training data to learn to recognize entities and relationships.

4.2.1 Aviation Safety Reports

Safety reports play a crucial role in data-driven safety oversight in safety domains like the aviation safety field. Although the content of the reports is typically highly informative, its transformation into a structured form, for example by means of dedicated reporting forms, is lossy and imprecise which negatively influences their potential for proper safety analyses. Text processing for such reports is essential for simplification of the safety reporting process.

4.2.1.1 Corpus

For this task, we had a set of incidents/ accidents reports in the field of aviation safety collected from different authorities' resources, for example, Air Accidents Investigation Institute (AAII)[7] in the Czech Republic, where they have their public aviation investigation incidents and accidents reports. Furthermore, the corpus contains confidential data provided by other study partners.

[7] http://www.uzpln.cz/

The seed ontology used to link entities is the Aviation Safety Ontology (ASO) introduced in [70] and available as an OWL 2 ontology[8].

4.2.1.2 Entity Linking Pipeline

Entity recognition tools that allow the use of background knowledge in the extraction process were tested separately, then the most accurate tools that served our purpose the best were chosen. We combined these tools in one pipeline to cooperate together.

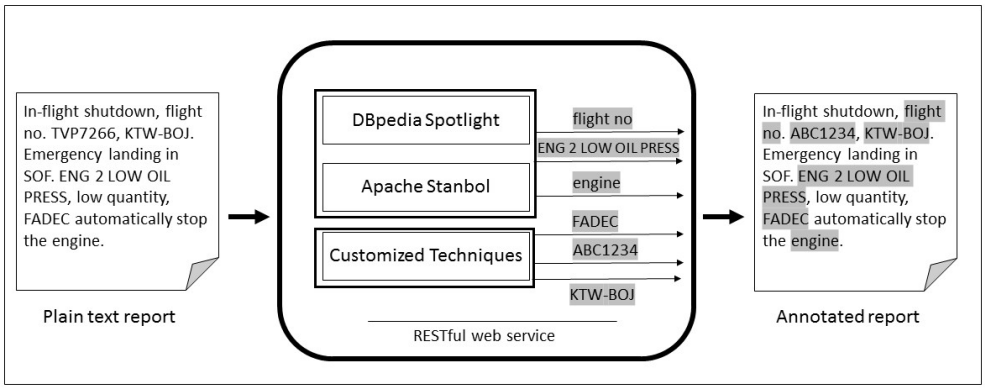


Figure 4.3: Aviation Safety Text processing and Annotation pipeline

As shown in Figure 4.3, DBpedia Spotlight[9] is an entity recognition tool that offers the possibility to create a spotlight model on the user's own server to model the occurrences of resources with the context in which they are mentioned. Building such a model requires a huge amount of pre-annotated data to train the model, which is not feasible in this case. However, the already-trained model can still extract entities based on the generic DBpedia ontology.

On the other hand, Apache Stanbol[10] is one of the tools that provide the ability to work with custom vocabularies and create custom indexes upon it. It also comes with a list of enhancement engine implementations, with the ability to build a specific one, based on a custom vocabulary, to get the most benefit out of the tool. This allowed building a chain of enhancement engines that fits perfectly for the detection of aviation safety concepts based on the Aviation Safety Ontology task.

We take into consideration the entities that are not possible for the current tools to detect, in spite of their ability to detect mentions from a specific termi-

[8]http://www.inbas.cz/aviation-safety-ontology
[9]https://github.com/dbpedia-spotlight/dbpedia-spotlight-model
[10]https://stanbol.apache.org

nology. For this complication that stems directly from the nature of the aviation domain, such as callsigns, registration marks, flight numbers, airport names abbreviations, etc., we use special techniques for every case as discussed in [71], [72] mainly following specific linguistic patterns. The output of Apache Stanbol, DBpedia spotlight, and the techniques for the special terms were parsed, merged, and optimized in a RESTful web service, and the mentions being detected with their proper mapping to the ontology hierarchy.

4.2.1.3 Results and evaluation

Different Linked Data Knowledge Extraction tools with respect to a domain-specific vocabulary had been tested, then we chose the tools that allow the best results of entity recognition, combining them into one pipeline, and making them work together, as well as with other features that we added, taking into consideration some very specific terms and abbreviations used in the aviation field.

This approach integrates several techniques in order to provide high-precision report annotations in the aviation safety domain in order to be used directly in practice.

Due to a noticeable fact in the aviation safety domain, many public safety reports are available. However, most of these reports are not-annotated or poorly automatically annotated, while very few reports are actually well-annotated. This makes the corpus construction process a very difficult task for the evaluation process, which requires extensive time and effort from the experts.

For evaluation purposes, we created rather a small, but high-quality, precise gold standard corpus out of, mainly initial aviation safety reports. Experts in the aviation domain manually annotated domain terms (entities) in each report with respect to the Aviation Safety Ontology (ASO), using the GATE tool[11].

This corpus consists of 80 high-quality annotated documents. This kind of corpus is needed for the annotation pipeline evaluation process using recall, precision, and F1 score metrics.

The precision scores high rates in most cases, it even reaches the 100% rate for some reports. On the other hand, the recall scores low rates.

In addition to the limitations of the tools to handle the Czech language, the tools do not support out-of-the-box further ontology learning techniques. For these reasons, we considered using a different set of tools in our next experiment.

4.2.2 Aircraft Reliability and Quality

Similar to the systems discussed in Section 4.2.1, documents such as aircraft manuals, audit, and maintenance reports provide a rich source of information

[11] https://gate.ac.uk/

supporting use cases related to the extraction of structured data from unstructured text documents for aircraft reliability and quality-related knowledge systems.

4.2.2.1 Corpus

The corpus is also taken from the aviation safety domain and is similar in nature to the previous experiment presented in Section 4.2.1. However, the documents are divided into two main categories. One is focused on the operational data or data describing specific events and states of the aircraft, and the other contains general information about aircraft parts, their functionality, and possible failures, e.g., Aircraft Operational Manuals.

The ontology used to link entities from textual documents is the Reliability and Quality ontology was developed using the Systematic Approach for Building Ontologies (SABiO) [73], which involves the use of an upper-level ontology, i.e. the Unified Foundation Ontology (UFO).

4.2.2.2 Entity Linking Pipeline

As already emphasized, performing NLP preprocessing tasks is essential to enhance textual documents and is a common task performed to prepare the required input for any entity linking system.

The extraction pipeline for the context of this experiment was implemented using General Architecture for Text Engineering (GATE) components. Gazetteers within the GATE framework refer to lists of (typically named) entities, e.g. the list of all countries or months of a year. They are essential components for performing the (named) entity recognition task. It is possible to build such gazetteers based on ontology as input, and GATE provides the possibility of this transformation using the OntoRoot Gazetteer and the Flexible Gazetteer, which we use to build the linking pipeline.

Detailed steps on how to configure the preprocessing pipeline and the basic setup for OntoRoot Gazetteer based on the input domain ontology can be found in Appendix A.

All produced annotations are of type Lookup, with additional features that give details about the resources to which they refer in the given ontology. More details about GATE gazetteers can be found on the official GATE documentation pages[12].

4.2.2.3 Results and Evaluation

Selected documents from the unstructured text corpus were manually annotated by domain experts to serve as the golden standard. The annotations are done

[12]https://gate.ac.uk/sale/tao/splitch13.html

according to the Reliability and Quality ontology which provides key categories of entities concerning reliability and quality.

The evaluation results showed that the GATE components were successful in most cases in linking the mentions of the entities with their corresponding identifiers in the Reliability and Quality ontology with an F1 score of 87%. As GATE does not support components and plug-ins that are capable of handling resources in the Czech language, it is interesting to test the linking system presented in 4.2.1 to compare the performance on the English documents and extend the work to cover the Czech part of the corpus in future work.

Chapter 5

Ontology Learning

In this chapter, we present a set of preliminary Lexico-Semantic Patterns (LSPs) to enrich the ontology with semantic entities extracted from relevant documents. The General Architecture for Text Engineering (GATE) framework provides Java Annotation Patterns Engine (JAPE), which is a language for expressing patterns to be used to extend the extraction pipeline, which makes GATE a good tool to extend the experiment carried out in Section 4.2.2, focusing mainly on learning relations between concepts, for example, the *part-whole* relationship from the available English textual documents. However, adapting GATE and JAPE for the Czech language is not possible as of the current state of the tool, since it does not provide a language model for processing documents and ontologies in the Czech language. To extend the pipeline introduced in Section 4.1 to process Czech resources (as a limited-resource language), we created a preliminary set of patterns written in the Hermes Information Extraction Language (HIEL) to learn common ontology relations from Czech documents, which we present in detail in the following section.

5.1 Limited-Resources Language

Even though the domain ontologies presented in Section 4.1 are rich, they are still far from complete. Updating the ontology manually is an exhaustive process, for that, it is crucial to support the process of developing the ontology with automatic suggestions to the user. Statistical information extraction does not provide satisfactory results when running on a small domain-specific corpus. We define a set of Lexico-Semantic Patterns (LSPs) to help the user build the ontology. Most of the research on LSPs is done on English documents. Only some attempts have been made in other languages such as French and German. To the best of our knowledge, no such work exists on Slavic languages as for Czech, which is crucial to support the given domain applications and users. In our case, we define a set of LSPs for the Czech language focusing on common

ontology relations.

5.1.1 LSPs - HIEL

For the definition of patterns, we use the Hermes Information Extraction Language (HIEL) that enables selecting concepts from the knowledge base and incorporating them into lexical patterns. HIEL patterns are an ordered collection of tokens that are divided by spaces. They are described by two parts, a Left-Hand Side (LHS) that defines the relation to be extracted and a Right-Hand Side (RHS) that describes the pattern that should be extracted from the text. Once the RHS has been matched in the text to be processed, it is annotated as described by the LHS of the pattern. Usually, the syntax of the pattern is denoted as follows:

$$LHS :- RHS$$

The language supports lexical characteristics such as a limited list of POS tags, concepts and relations, literals, logical operators (and, or, not), repetition operators (*, +, ?), and wildcards (%, _). We extended the restricted symbols and abbreviations of the lexico-syntactic pattern used in [74]. The list of the abbreviations and common lexical categories used to formalize our patterns can be found in Table 5.1.

In our experiments and with the help of domain experts, we performed a linguistic analysis and manually defined a preliminary set of LSPs corresponding to Ontology Design Patterns (ODPs) that captures basic ontology relations, such as *subClassOf*, *equivalence*, *part-whole*, and *hasProperty* relations.

In the following patterns, the Left-Hand Side for the rules is represented as:

$$LHS = (\$subject, relationOfInterest, \$object)$$

In Tables 5.2, 5.3, 5.4, and 5.5, we present only the Right-Hand Side part of the rules due to the space presentation limit. We also provide examples extracted from our data.

Table 5.1: LSPs symbols and lexical categories

Symbols & Abbreviations	Description & Examples
CATV	Phrases of classification. For example, rozlišuje (distinguishes), člení se (is divided into), etc.
COMP	Phrases of composition. For example, zahrnuje (includes), tvořený (formed), skládající se (consisting of), členěno na (divided into).
COMPR	Phrases of reverse composition. For example, vyskytující se v (appearing in), tvoří (creates), je součástí (is part[of]).
CN	Phrases of generic class names. For example, základní typy (base types of).
SYN	Phrases of synonyms. For example, ekvivalent (equivalent).
PROP	Phrases of properties. For example, je přiřazen (is attached).
BE, CD, DT	Verb to be, Cardinal number, Determiner, respectively.
NN, JJ, RB, IN	Noun, Adjective, Adverb, Preposition, respectively.

Table 5.3: LSPs corresponding to part-whole rules

P_{id}	**RHS**
P_{21}	$\$subject : Concept\ COMP\ RB?\ IN?\ \$object : Concept$
	example: **Správní území Prahy** členěno na **lokality**.
	meaning: Administrative territory of Prague is divided into localities.
P_{22}	$\$subject : Concept\ COMPR\ IN?\ \$object : Concept$
	example: **Veřejná prostranství** tvoří **ulice**.
	meaning: Public areas are created by streets.

Table 5.2: LSPs corresponding to subClassOf rules

P_{id}	**RHS**
P_{11}	$CATV\ CD\ CN\ \$object : Concept\ DT?\ \$subject : Concept$ $CATV\ CD\ CN\ \$object : Concept\ DT?\ Concept\ (\ 'a'|',')\ \$subject : Concept$
	example: Metropolitní plán rozlišuje dva základní typy **krajin městskou** a **otevřenou**.
	meaning: Metropolitan plan distinguishes two base types of landscape: municipal landscape and open landscape.
P_{12}	$\$object : Concept\ IN?\ CATV\ IN?\ \$subject : Concept$ $\$object : Concept\ IN?\ CATV\ IN?\ Concept\ (\ 'a'|',')\ \$subject : Concept$
	example: **Parkem** [se rozumí] vymezená část území s rozlišením na **městský park** a **krajinný park**.
	meaning: Park [is understood as] delimited part of area, further distinguished into municipal park and landscape park.
P_{13}	$\$subject : Concept\ BE\ \$object : Concept$
	example: **Metropolitní plán** je především **plánem** struktury území.
	meaning: The metropolitan plan is primarily a plan of the area structure.

Table 5.4: LSPs corresponding to equivalence rules

P_{id}	**RHS**
P_{31}	$\$subject : Concept\ BE?\ SYN\ NN?\$object : Concept$ $\$subject : Concept\ BE?\ SYN\ NN?\ Concept\ ('a'\ |\ ')\ \$object : Concept$
	example: **Metropolitní** je ekvivalentem pojmů **celoměstský** a **nadmístní**.
	meaning: Metropolitan is equivalent of terms citywide and supralocal.
P_{32}	$\$subject : Concept\ DT?\ SYN\ DT?\ \$object : Concept$
	example: **Krajinou za městem**, syn. **krajinným zázemím města**.
	meaning: Landscape outside the city, synonym. city landscape background.

Table 5.5: LSPs corresponding to hasProperty rules

P_{id}	**RHS**
P_{41}	$\$subject : (Concept \mid (JJ?\ NN?))\ BE\ PROP\ \$object : (Concept \mid (JJ\ NN) \mid NN)$
	example: **Každé lokalitě** je přiřazen **typ struktury**.
	meaning: Every locality has assigned type of structure.
P_{42}	$CD\ CN\ Concept\ IN?\ \$subject : Concept\ DT?\ CD?\ \$object : Concept$ $CD\ CN\ Concept\ IN?\ \$subject : Concept\ DT?\ CD?\ Concept\ ('a' \mid ',')\ CD?\ \$object : Concept$
	example: Deset typů struktur pro zastavitelné stavební lokality: (01) rostlá struktura, (02) bloková struktura,...
	meaning: Ten types of structures for buildable localities are (01) growing structure, (02) block structure,...

5.1.2 Evaluation and Discussion

We evaluated the patterns defined in Section 5.1.1 on the same textual documents that are annotated and parsed by Annotace as described in Section 4.1.1. Domain experts provided their approval or rejection of the new relations extracted from the annotated documents after applying the patterns. The patterns achieved an average of 65% precision, 57% recall, and an F1 score of 61%. Table 5.6 gives a closer look at the precision and recall achieved by each pattern.

The false negative cases occurred mainly when the phrase was not recognized in the text as a term occurrence, and hence the sentence did not match the specified pattern. For this reason, we extend the patterns to extract the subject or the object as the noun or the combination of adjective-noun. This improved the performance of the patterns and helped to recognize more terms that were not retrieved by Annotace. On the other hand, some patterns suffered from the over-generating problem.

The challenge of the free-word order of the Czech language that leads to inverse relation explains many cases where false positives were encountered. For example, pattern P_{12} was able to extract the two sides of the *subClassOf*

Table 5.6: Lexico-semantic patterns evaluation in terms of precision and recall

	Precision	Recall
P11	76%	40%
P12	51%	54%
P13	63%	60%
P21	74%	70%
P22	69%	53%
P31	78%	81%
P32	83%	75%
P41	85%	87%
P42	80%	56%

relation correctly but wrongly reversed the assignment of the super-class and the sub-class in some cases. A possible solution is to consider the case of the words in addition to their position. Unfortunately, we could not investigate further because the Hermes language allows only the usage of specific tags. However, the free-word order problem of the Czech language is a challenge even after considering syntactic information. The problem is that, for example, the nominative case is similar to the accusative case when the noun is plural in some situations. This would make it difficult for even an expert to get the relation correctly based only on ambiguous syntactic information. Consider the sentence, *"Zastavitelné území tvoří plochy zastavitelné"* (en. "Buildable area creates buildable surfaces") which represents exactly this case where the verb "tvoří" can be used in both directions, and "zastavitelné území", and "plochy zastavitelné" will have the same form in the nominative and accusative linguistic cases.

The type of recognized relation is another open issue. Pattern P_{21} wrongly retrieved concepts that had a *hyponym-hypernym* relation as a *part-whole* relation. This happens when a word that, according to our experts, intuitively refers to a *part-whole* relation but is used in the text carelessly. Another common issue we found in the data is that the text does not always provide complete information to be extracted. For example, for the sentence *"Metropolitní plán rozlišuje stanici metra, vestibul stanice metra a depo metra."* (meaning Metropolitan plan distinguishes subway station, subway station lobby and subway depot), pattern P_{12} extracted **"Stanice metra"**, **"Vestibul stanice metra"** and **"Depo metra"** to be sub-classes of **"Metropolitní plán"**. However, this is not the case since **"Metropolitní plán"** is the term used to represent the document itself and hence, the extracted terms are sub-classes of a super-class that is not mentioned in the text.

Patterns P_3 and P_4 achieved reasonably high scores. However, only a few instances were found in the corpus.

5.2 Limited Training Data

Apache Stanbol and DBpedia OBIE tools used in Section 4.2.1 to extract information from aviation incident and accident reports based on the ASO ontology do not provide out-of-the-box support for further enhancing the ontology. Furthermore, the Apache Stanbol project has been retired[1] and no longer is supported. For these reasons, we did not carry out further experiments on this pipeline.

On the other hand, to extend the Aircraft Reliability and Quality ontology based on the annotated output documents in Section 4.2.2, it is possible to define LSPs using JAPE rules, taking advantage of the linguistic preprocessing components provided by the GATE framework that are suitable for resources in the English language. For example, this step can extract new components, systems, or relations between entities, such as revealing a part-whole relation between two airplane components, using the advantage of the annotated text produced in the previous step in the pipeline as described in Section 4.2.2.2, together with the ontology, by writing suitable JAPE patterns to serve this purpose. However, it is needed to first make some changes to the annotations in order to create Ontology-aware JAPE rules.

5.2.1 LSPs - JAPE

Java Annotation Patterns Engine (JAPE) patterns are, in general, an ordered collection of tokens. They are described by two parts, a LHS that defines search expressions on the annotated text and a RHS of the rule that contains the statements that manipulate and create annotations. Details about basic JAPE operations, including equality, comparison, contextual operators, and regular expressions, and tools, including features and values, meta-properties, sequences - alternatives and grouping, ranges, multi-constraint statements, negation, and repetition can be found in the official JAPE documentation[2]. Note that any annotation to be defined in the LHS of the rule must be included in the input header. This means that any annotation that is not included in the input header will be ignored (e.g. whitespace). Using this feature, it is possible to add the annotation type “Split” into the input header to limit matching so that it is done only within the sentence boundary, i.e. on the sentence level.

```
Phase: ...
Input: Split ...
Options: ...
Rule: ...
```

[1] https://attic.apache.org/projects/stanbol.html last accessed 2023-05-29
[2] https://gate.ac.uk/sale/tao/splitch8.html

General notes on the experience acquired when creating JAPE patterns and ontology-aware JAPE can be found in Appendix B.

In the JAPE ontology-aware mode, the matching between two class values will not be based on simple equality, but rather on hierarchical compatibility. For example, if the ontology contains a class named *Wing*, which is a subclass of the class *Component*, then a pattern of *Entity.class* == '*Component*' will successfully match an annotation of type Entity with a feature class having the value 'wing'. The following rule will create a new type of annotation called 'Component'. It will annotate all mentions of the *Component* class, its subclasses, or any instances under that class.

```
Phase: OntoMatching
Input: Lookup
Options: control = appelt
Rule: ComponentLookup
(
  {Lookup.class == Component}
 ):component
-->
:component.Component= {class = :component.Lookup.class,
                       inst = :component.Lookup.inst}
```

In the following, we show the implementation of the *hasComponent* rule between two lookup annotations of type *Component*. Let us take the following text input examples.

The two ***main spars*** *and both* ***nacelles*** *are part of the* ***center wing****.*
An aluminum ***fuel tank*** *is installed in each of the* ***wings****.*
The ***wings*** *have a* ***front*** *and* ***rear spar****.*
Each ***wing*** *has a* ***top shell*** *and a* ***bottom shell****.*
Both the ***stabilizers*** *have* ***twin spars****.*

The tokens in bold are the Lookups (mentions of the ontological entities) found during the entity linking phase; see Section 4.2.2.2. The *italic* ones are special phrases that indicate a relationship of interest. To model these phrases, we use similar definitions in Table 5.1. In the following rule, COMP is a special class representing phrases of composition that indicate the partonomy relation. For example, have, compose, consist, etc. Based on the context, we define the class COMPR which indicates the phrases of reversed composition, where the whole appears later in the sentence, for example, part, install, etc. Similarly, it is possible to define other classes like CATV representing phrases of classification such as categorize into, classify into, etc. Simple implementation of the JAPE rule to reveal the part-whole relation of interest, matching, in one sentence, any sequence, is as follows.

```
Superconcept COMP subconcept1 subconcept2?

Phase: OntoMatching
Input: Split Lookup
Options: control = appelt

Rule: HasComponentRelation1
(
  ({Lookup.class == Component}):superconcept
  {Lookup.class == "http://onto.fel.cvut.cz/ontologies/
                   lexicon/COMP"}
  ({Lookup.class == Component}):subconcept1
  ({Lookup.class == Component})?:subconcept2

):hasComponent1
-->
:hasComponent1.Relation= {uri = "http://onto.fel.cvut.cz/
ontologies/fmea/hasComponent",
rule = "HasComponentRelation1",
Arg1 = :superconcept.Lookup.class,
Arg2 = :subconcept1.Lookup.class,
Arg3 = :subconcept2.Lookup.class}
```

While implementing the JAPE rule, it is possible to combine syntactic (e.g. the category of the token generated by the part of speech tagger component in the pipeline) or other types of annotations (e.g. NounChunk) as well as semantic knowledge (generated by the ontoRoot gazetteer component in the pipeline) in the body of the rule to form the lexico-semantic pattern. For example, the previous rule can be more specified as follows,

```
Superconcept COMP subconcept1 [,] CC subconcept2?

Phase: OntoMatching
Input: Token Lookup
Options: control = appelt

Macro: LIST2
(
(({Token.category == DT})?
({Lookup.class == Component}):sub1)
({Token.string == ","})?
```

```
({Token.category == CC})
(({Token.category == DT})?
({Lookup.class == Component}):sub2)
)

Rule: COMPRelation1
(
  ({Lookup.class == Component}):superconcept
  ({Lookup.class == "http://onto.fel.cvut.cz/ontologies/
  lexicon/COMP"})
  (LIST2):subconcept

):comp1
-->
:comp1.Relation= {uri = "http://onto.fel.cvut.cz/
ontologies/fmea/hasComponent", rule = "COMPRelation1",
Arg1 = :superconcept.Lookup.class,
Arg2 = :sub1.Lookup.class, Arg3 = :sub2.Lookup.class}
```

5.2.2 Evaluation and Discussion

Since GATE and JAPE cannot adapt to Czech resources, we limited experimenting with only a small subset of patterns as a proof of concept of the general approach, since we are dealing with multilingual input documents and need to provide support for processing all input languages, including Czech.

The set of patterns was able to extract the part-whole relationship from the text with an F1 score of 87% compared to the extracted relations in the golden standard annotated by experts.

Fine-tuning the set of patterns can further improve this score. Therefore, we found the pattern-based approach very efficient given the limited resources to learn the ontology from multilingual input documents. A future consideration is to adapt the pipeline presented in Section 5.1 to implement the patterns in the HIEL language.

5.3 Ontology Learning Enhancement

Textual documents, for example, aviation safety reports usually contain different types of data that can enhance the understanding of these documents in different aspects. Temporal knowledge and geographical knowledge are examples of such data, and it is important to be able to consider extracting this

knowledge as well to enhance the domain ontology and improve its accuracy and completeness. For example, for the Aviation Safety Ontology, textual documents can be further analyzed to extract information about the location and time of aircraft incidents or accidents. This information can be used to include temporal and spatial dimensions in ontology classes. The same applies to urban planning and development-related ontologies, where information about location, size, and type of buildings or infrastructure projects can be extracted.

We carry out our next experiment on extracting temporal information from larger datasets in Linked Open Data (LOD) to describe datasets temporally. LOD datasets are described temporally by DCAT vocabulary [75], for example, which forms a comparison baseline to evaluate the temporal information extraction pipeline described in the next section. However, ontologies can be considered as datasets, as in TermIt for example, so the approach is applied to extend any ontology with the temporal dimension.

5.3.1 Temporal Information Extraction

Due to the fact that temporal expression extraction is a subtask of the general information extraction task and it needs a special dataset for performing experiments and enriching the current solutions, we performed our next experiments on a general domain focusing on extracting only temporal data from linked data cloud to reconstruct higher-level knowledge compliant with current temporal ontologies.

As structured data found in the LOD typically do not have an explicitly defined schema, the notion of dataset summaries emerged in the context of Linked Data. This notion stems from the dataset exploration scenario in which datasets need to be described in order to be discovered. In [76], dataset descriptors have the form of datasets describing other datasets, typically represented as metadata. We approach the problem of exploring datasets temporally where we describe the datasets with their temporal information, which allows to equip various parts of the dataset with time instants/ intervals that the actual content of the dataset speaks about. A temporal value or set of temporal values, which we call the time scope of the statements, is associated with each temporal statement that describes its temporal extent. We compute the temporal coverage of the dataset and formalize it using an integration between two ontologies, the Unified Foundational Ontology (UFO-B) [9] and the OWL Time Ontology [12], which we call the Temporal Descriptor Ontology (TDO).

Regarding temporal information understanding in the current state-of-the-art, there are missing connections between existing extraction techniques and the reality of the data in the LOD. Data in LOD use particular formats and knowledge. Extracting temporal information should not only be about applying

NLP techniques but also taking into consideration the RDF knowledge, the connections and relations that already exist, and the typology and structure of the data within LOD. We approach this matter by contextualizing current extraction techniques and directing them to align with the ontology.

5.3.1.1 Temporal Data Analysis in the Linked Open Vocabularies

Vocabularies provide the semantic glue enabling Data to become meaningful Data. Linked Open Vocabularies (LOV)[3] offer several hundred of such vocabularies[4] frequently used in LOD, together with their mutual interconnections. A vocabulary in LOV gathers definitions of a set of classes and properties.

To offer a general solution, extensible towards most properties used across the LOD, we experimented with the most commonly used vocabularies in the datasets according to LOV focusing mainly on the vocabularies that are reused in the Czech Linked Open Data. In order to perform the temporal analysis and to distinguish the temporal properties describing the actual content of the datasets, two annotation properties were created:

is-temporal denotes data properties with an explicitly temporal data type expressing time or date, for example, but not limited to `xsd`[5]`:date`, `xsd:time`, `xsd:gYear`, etc.

has-temporal-potential to indicate data properties that potentially contain temporal knowledge in an unstructured form, usually expressed using natural language, for example, but not limited to `dcterms`[6]`:title`, `dcterms:description`, etc..

We experimented with the following vocabularies as they are most frequently reused in the Czech LOD: Schema.org[7], dcterms, prov-o[8], and goodRelations[9]. An ontology combining these vocabularies augmented with *is-temporal* and *has-temporal-potential* annotation properties is available[10] for further details. We notice that some properties accept multiple data type ranges, for instance *schema:temporalCoverage*, which accepts either DateTime, Text, or URL. Another observation is that *dcterms* vocabulary does not have any explicitly temporal data type. Even when the property aims to express only time or specific date, it is modeled to accept general literal[11] ranges.

[3] `http://lov.okfn.org/dataset/lov`
[4] currently 603 vocabularies, last accessed 27.07.2017.
[5] xsd: http://www.w3.org/2001/XMLSchema#
[6] http://purl.org/dc/terms/
[7] `http://schema.org/`
[8] `http://www.w3.org/ns/prov#`
[9] `http://purl.org/goodrelations/v1`
[10] available at `https://kbss.felk.cvut.cz/ontologies/dataset-descriptor/temporal-properties-model.ttl`
[11] `http://www.w3.org/2000/01/rdf-schema#Literal`

Table 5.7: Temporal knowledge in the Czech LOD

	is-temporal		has-temporal-potential	
	number of properties	number of triples	number of properties	number of triples
dcterms	12	7955431	2	7734043
schema.org	8	512781	4	149339
goodRelations	6	3110226	0	0
prov-o	2	146115	0	0
CzLOD	36	15268994	8	2597668

5.3.1.2 Analysis of the nature of temporal data in Czech LOD

Czech Linked Open Data cloud (Czech LOD)[12] contains dozens of datasets with approximately 1 billion records. We attempted to investigate the nature of temporal knowledge. We manually experimented with a test set of 10 datasets found in the Czech Linked Data Cloud involving datasets published by the *OpenData.cz* initiative, to reveal the nature of temporal knowledge taking into consideration the variety of topics and the contexts of the datasets. Exploration of the temporal structure in the graph is done through a set of SPARQL queries. We build the graph out of the data properties used within the datasets in a structured temporal form or the temporal knowledge found in the string literals. After analyzing the nature of the properties used in each dataset, we identify two types of temporal knowledge; *is-temporal* denotes the properties that represent time explicitly through a temporal data property, and *has-temporal-potential* which denotes the properties that contain temporal knowledge in the form of string literals which are usually expressed as short to intermediate length texts expressed in natural language. The latter type is where we will perform the analysis to reveal hidden temporal knowledge in the datasets.

The data inside the Czech LOD are heterogeneous. Triples contain string literals in English as well as in Czech, which makes the available temporal analysis tools not efficient enough. Additionally, temporal information varies from one dataset to another. In one dataset, usually only structured temporal information or only unstructured temporal information can be found. However, most of the datasets have both types of temporal information.

Based on the analysis of the temporal information in the datasets, we could recognize several forms. Time is mentioned explicitly or implicitly. Explicit time mentions have two possibilities, an instant, which is a precise time point, and an interval, which is a time period with distinct start and end points. However, an instant form of temporal information can be understood as an interval with the

[12] `http://linked.opendata.cz/`

same start and end time. Implicit time information found in the datasets has the form of mentions with a specific common well-known implicit temporal property (Christmas, Mother's Day, Independent Day). Although, this information can still differ according to the spatial information.

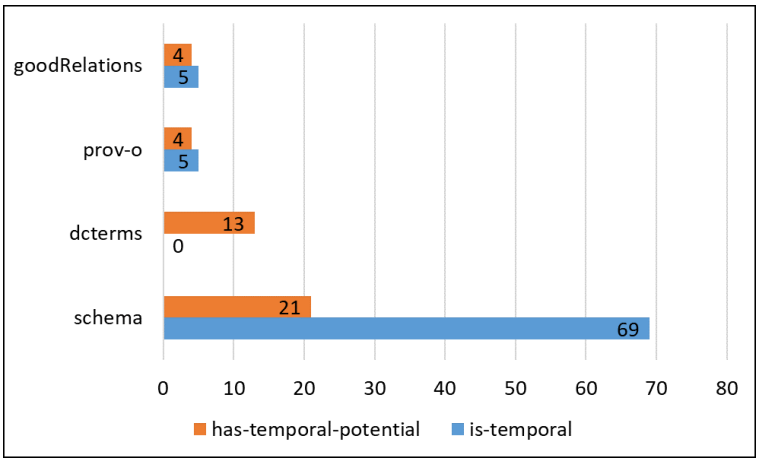


Figure 5.1: Temporal data-properties in the vocabularies

Figure 5.1 shows the overall number of annotated properties that describe temporal knowledge in the selected common vocabularies compared to the properties of temporal data that are reused across datasets in the Czech cloud. It is remarkable that certain temporal data properties in the vocabularies are completely reused in the datasets in the cloud. This gives the motivation to perform a wider analysis of the vocabularies properties in further work.

5.3.1.3 Extraction Pipeline

Temporal information extractors (namely *SUTime* and *Heidel Time*) are used in order to extract temporal knowledge from the string literals in the datasets.

Though SUTime has many advantages and can detect a wide spectrum of temporal knowledge, we observed many gaps in applying these tools to string literals. For example, missing the time range in sentence *"in the period of the 17th - 20th centuries"*, and the loss of day granularity of the detected date in sentence *"30. September 2002"*. This lack stems directly from the nature of the temporal data within the datasets as well as the lack of understanding of the context around. Also, data heterogeneity regarding the language used within the datasets.

To overcome some of these limits, we extend the rules in the rule files of the extraction tools with more definitions to detect the temporal expressions that were not possible to be detected by the default settings of the tools. In this way, we were able to increase the recall of the retrieved temporal knowledge. The following, respectively, are samples of the rules that we created to spot expressions like { *05/11/1999*},{ *05-11-1999*}, and even the special cases to

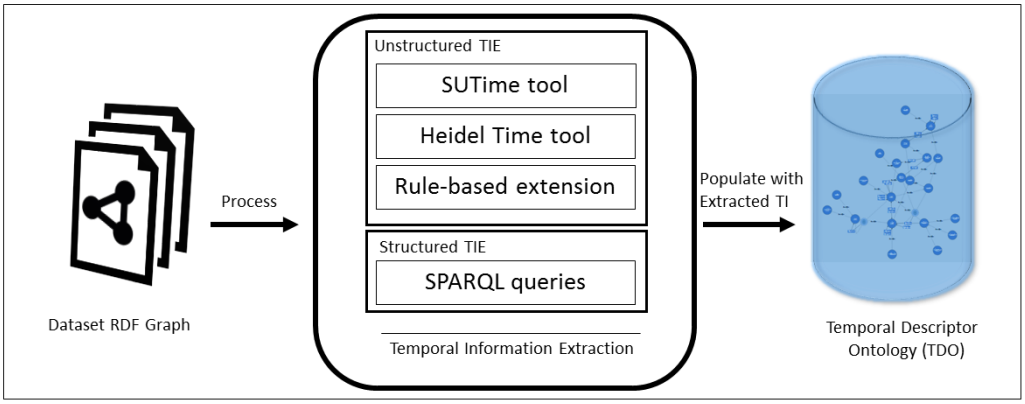


Figure 5.2: The scenario of extracting temporal information and populating the time ontology

extract the year from the data like $\{253/1992\ Sb\}$.

$$\{ruleType : "time", pattern : /dd/MM/yyyy/\}$$

$$\{ruleType : "time", pattern : /dd - MM - yyyy/\}$$

$$\{ruleType : "time", pattern : /[0-9]\{1,4\}.?yyyy'Sb'/\}$$

The analysis of the data within the datasets showed which features are needed to describe the nature of the time information. For example, if this temporal information is connected to a person, it means that it has a relationship, for example (date of birth, date of death, etc.). This information is needed to understand the nature of temporal knowledge, so it is not only needed to detect abstract time intervals and instances, but also to understand the context of this temporal information and its meaning, which is a piece of important information to consider. This approach is designed to provide a better representation, and hence better temporal exploration experience of the datasets.

5.3.1.4 Temporal Descriptors Ontology

Based on the analysis in the previous section, in order to describe datasets based on temporal knowledge and achieve a better representation of their temporal dimension, we want to create an integrating ontology on top of the common vocabulary. To grasp the reality, the Temporal Descriptor Ontology (TDO) is built on top of the Unified Foundation Ontology (UFO), which is one of the top-level ontologies that has a good modeling language and is supported by several useful tools. UFO presents a level of abstraction that forms a perfect point to start building the TDO. Namely, UFO-B, which is used to model the extracted events, whether they are atomic or complex, the participants of the events, and the time span within which the events occur. UFO-B suggests that since events happen in time, they are framed by a **Time Interval** [77], but the

provided representation of temporal knowledge is very limited.

At this point, we extend the UFO-B with the temporal data captured by the *OWL-Time Ontology* and connect it to the events. This provides a generic framework for extracting temporal information from the datasets. The core parts of the TDO are shown in Figure 5.3.

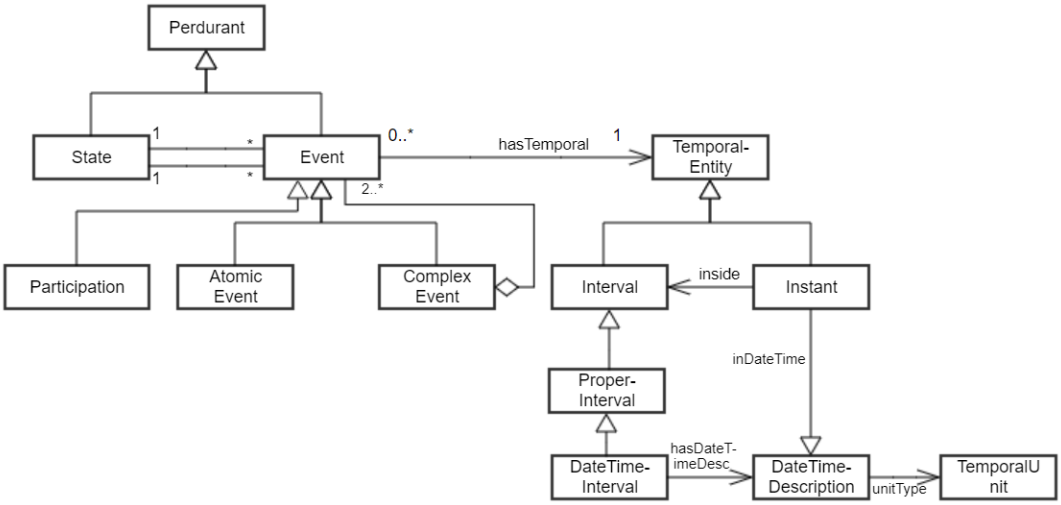


Figure 5.3: Temporal Descriptor Ontology

The example in Figure 5.4 shows the representation of an extracted event and is the temporal interval for the year 1996 using the properties *time:hasBeginning* and *time:hasEnd*.

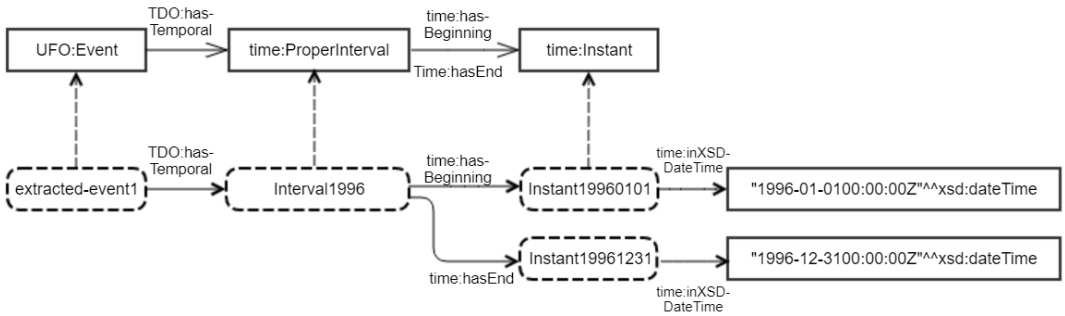


Figure 5.4: Temporal Descriptor Ontology example

5.3.1.5 Results and evaluation

DCAT vocabulary provides the property *dct:temporal* to annotate datasets with temporal coverage. According to DCAT specifications, it describes the temporal period that the dataset covers. We used this property, as well as the temporal descriptors presented in [76] to investigate the quality of the temporal metadata based on comparison with temporal information extracted from the actual content of the datasets.

We computed the temporal representation of the datasets in the Czech cloud using the approach we presented in the previous section. We compute the

Table 5.8: Comparison of temporal coverage by DCAT metadata, temporal descriptor (TD), and the actual content (AC) temporal representation computed using our approach. *Missing temporal DCAT metadata are indicated by empty cells within the 2nd and 3rd columns. Empty cells in the 4th and 5th columns might indicate either missing data or an incomplete descriptor computation procedure. Complete computation of the temporal coverage can be found in the 6th and 7th columns.*

Dataset	DCAT startDate	DCAT endDate	TD minDate	TD maxDate	AC minDate	AC maxDate
ds:coi.cz/kontroly	2012-01-01	2014-12-31	2012-01-02	2014-12-31	2012-01-02	2014-12-31
ds:coi.cz/sankce	2012-01-01	2014-12-31	2012-01-01	2014-12-31	2012-01-01	2014-12-31
ds:currency			1999-01-04	2016-04-29	1999-01-04	2016-04-29
ds:drugbank			0002-02-26	2030-12-31	1633-01-01	2030-12-31
ds:legislation/psp.cz			0014-01-01	2099-01-01	0014-01-01	2099-01-01
ds:political-parties-cz			1928-04-12	2013-12-18	1928-04-12	2013-12-18
ds:seznam.gov.cz/agendy			2011-11-16	2030-12-31	1939-01-01	2030-12-31
seznam:objednavky	2014-01-01	2015-02-26	0201-10-07	2020-08-19	2000-01-01	2020-08-19
seznam:plneni			2013-03-26	2015-05-07	2013-03-26	2016-03-31
seznam:smlouvy	2014-01-01	2015-02-26	1992-08-25	2015-05-22	1945-01-01	2019-12-31
ds:vavai/evaluation/2009	2009-01-01	2009-12-31			1034-01-01	2030-12-31
ds:vavai/programmes	2015-03-30	2015-03-30	2003-02-12	2015-01-27	1959-01-01	2022-12-31
ds:vavai/research-plans	2015-03-30	2015-03-30	2003-01-01	2014-06-30	1890-01-01	2014-06-30
ds:vavai/tenders	2015-03-30	2015-03-30	1995-10-02	2016-04-01	1995-10-02	2022-12-31
ds:pravni-vztahy-listiny	2015-02-26	2015-02-26	1993-01-01	2015-07-16	1950-12-31	2015-07-16

temporal scope of the actual content of the triples in the datasets. Table 5.8[13] shows the overall temporal coverage datasets in the Czech cloud that have both temporal descriptor representation as presented in [76], computed by our approach and compared to the temporal coverage computed by DCAT and the temporal descriptors. Each line represents a dataset in the Czech cloud. The second and third columns represent the temporal coverage defined by the DCAT property dct:temporal[14]. The next two columns represent the minimal and maximal calculated date by the temporal descriptor. The last two columns represent minimal and maximal temporal extractions that we computed from the actual content of the dataset using the pipeline presented in 5.3.1.3.

For a test pad of 15 datasets, the temporal representation computed using our approach is compatible with the temporal descriptors for 46.6% of the cases. For 46.6% of the cases, they differ due to the fact that our approach takes the unstructured temporal information into consideration, while the temporal descriptors care only about the temporal meta-data in the dataset. For the same reason, the dataset *ds:vavai/evaluation/2009* does not have any temporal descriptor representation, while using our approach, we are able to compute the temporal coverage for this dataset.

Next, we extend the experiments and utilize our approach to compute the temporal description of all the datasets available in the SPARQL endpoint of the Czech cloud which contains 76 datasets[15]. We are able to augment the temporal representation for 57.89% of the datasets. The rest of the datasets do

[13]Each dataset in the table prefixed with *"ds"* representing URL `http://linked.opendata.cz/resource/dataset/`; Each dataset in the table prefixed with *"seznam"* representing URL `http://linked.opendata.cz/resource/dataset/seznam.gov.cz/rejstriky/`.

[14]`http://purl.org/dc/terms/temporal`

[15]last access 2017-08-05

Table 5.9: The complementary comparison of the temporal coverage by DCAT metadata, to the *actual content* temporal representation computed using our approach for the **rest** of the datasets in the Czech cloud. *Missing temporal DCAT metadata are indicated by empty cells within the 2nd and 3rd columns.*

Dataset	DCAT startDate	DCAT endDate	AC minDate	AC maxDate
ds:ic	2015-01-01	2015-02-26	1972-01-01	2013-12-20
ds:mfcr/ciselniky	2010-01-01	2014-12-31	1900-01-01	9999-12-31
ds:coi.cz/zakazy	2012-01-01	2014-12-31	1986-01-01	2001-01-01
ds:vavai/evaluation/2013	2013-01-01	2013-12-31	1277-01-01	2016-12-31
ds:sukl/drug-prices	2012-01-01	2015-07-29	1990-01-01	2016-04-20
ds:cenia.cz/irz	2015-02-01	2015-03-31	2004-01-01	2012-12-31
ds:vavai/funding-providers	2015-03-30	2015-03-30	1996-12-31	2007-06-01
ds:vavai/evaluation/2011	2011-01-01	2011-12-31	1100-01-01	2050-12-31
ds:vavai/evaluation/2010	2010-01-01	2010-12-31	0900-01-01	2050-12-31
ds:check-actions-law			1945-10-27	2013-12-31
ds:vavai/results	2015-02-26	2015-02-26	1970-01-01	2020-12-31
ds-external:pomocne-ciselniky			1988-01-01	2010-12-31
ds:vavai/evaluation/2012	2012-01-01	2012-12-31	1100-01-01	2050-12-31
ds:vavai/projects	2015-03-30	2015-03-30	1100-01-01	2050-12-31
ds-external:check-actions			1992-01-01	2016-08-01
ds:it/aifa/drug-prices	2012-01-01	2015-07-31	2015-07-15	2015-07-15
ds:/court/cz			2013-06-17	2013-06-17
ds:nci-thesaurus			2013-03-25	2013-05-15
ds:obce-okresy-kraje			2012-09-05	2012-09-05
ds:cpv-2008			2008-01-01	2008-01-01
ds:spc/ai-interactions			2013-05-15	2013-05-15
ds-external:nuts2008/			2008-01-01	2011-12-31
ds-external:geovoc-nuts			2013-01-04	2013-01-04
ds:dataset/fda/spl			2013-15-05	2013-15-05
ds:vavai/cep			1141-01-01	2020-12-31
ds:regions/momc			2014-02-24	2014-02-24
ds:buyer-profiles/contracts/cz			2000-01-01	2024-12-31
ds:legislation/nsoud.cz			2004-01-27	2014-02-27
ds-external:souhrnné-typy-ovm			1969-01-01	2012-12-31

not have any temporal knowledge in their resources to be extracted.

Table 5.9[16] shows the computed temporal coverage (minimum and maximum date extracted) of each dataset in the cloud compared to the temporal coverage of DCAT when available.

The dataset **ds:vavai/evaluation/2011** contains data about events that started during *the 12th century* (e.g., Pilgrimage element in crusades with Czech participation in the twelfth century). For that, the actual data coverage starts at *1100-01-01*. On the other hand, the dataset **ds:vavai/projects** maximum coverage date is *2050-12-31* (Prediction processing of systems utilizing renewable energy sources in the Czech Republic till 2050).

[16]All the dates are normalized to a day granularity.

Chapter 6

Applications and Use Cases

This chapter presents applications and dedicated use cases in which entity linking, ontology learning, or a combination were used in an end-to-end pipeline.

6.1 Semantic Vocabulary Manager - TermIt

TermIt[1] is an integrated system for managing a set of interconnected vocabularies, identification of individual concepts in source documents and interlinking them, and using such terms for semantic data asset annotation and subsequent search [78].

TermIt supports regular vocabularies in addition to the so-called document vocabularies. Document vocabularies are associated with a document that may consist of several files, i.e., a small set of files forming one document. The document vocabulary is based on a normative document whose text presents the source of the terms and terms definitions in the vocabulary. Therefore, it is an ideal use case for the rule-based information extraction and ontology learning approach. The two main functional areas of TermIt are *vocabulary management* and *resource annotation and search*.

Creating Vocabularies Based on Documents - This use case represents a situation where the user wants to create a vocabulary based on a document. The user can create a *document-vocabulary* and uploads the relevant file(s) to TermIt. Afterwards, it is possible to run an automated text analysis service on these files. This service is able to suggest new terms based on their significance in file content, helping the user to start building the *seed ontology*, or a *document-vocabulary* in the context of TermIt. The text analysis service will be discussed in more detail in Section 6.1.1. The user can review the suggestions made by the preliminary text analysis, create terms from them, or mark and create new terms manually. The annotated terms in the document will then automatically be created as document-vocabulary terms.

[1] `https://github.com/kbss-cvut/termit` accessed: 2023-05-29

Resource Annotation - This scenario assumes that a non-empty vocabulary already exists. This vocabulary can be used to annotate the resources registered in TermIt. The text analysis service can also discover *mentions* of terms in the text. The service suggests the mentions of the vocabulary terms, and the user may approve or discard them.

This implementation focuses on the Czech language with prospective usage for a larger class of languages, for example, Slavic ones. We also provide an implementation for the English language as an example of how to adapt the approach to other languages.

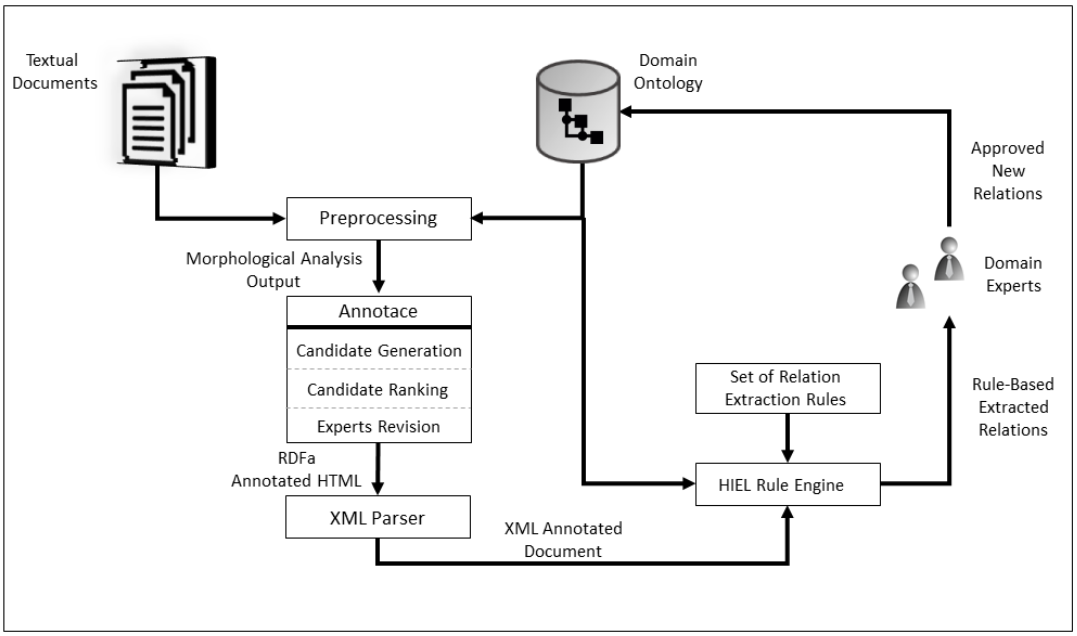


Figure 6.1: Entity linking and relation extraction pipeline

The implementation of the iterative approach presented in Section 3 is illustrated in Figure 6.1.

6.1.1 Annotace Implementation

As part of the processing stack and based on the entity linking pipeline presented in Section 4.1.2, **Annotace**[2], an entity linking service, was implemented and used in the context of TermIt, a terminology management tool based on Semantic Web technologies developed at Czech Technical University in Prague. TermIt allows managing vocabularies and documents that use terms from the vocabularies. The documents can be imported into the TermIt document manager and associated with vocabulary. The vocabulary can be empty or already augmented with some classes and instances. TermIt allows users to create and manage vocabularies based on related resources, and the entity linking service helps to automate this process in two scenarios:

[2]Source code is available at `https://github.com/kbss-cvut/annotace` accessed: 2023-05-29

- In the first scenario, a new document is uploaded into the TermIt document manager, and a newly created vocabulary is associated with it. The vocabulary is empty at this point. The task is to help the user to start building the vocabulary based on the text present in the document. Annotace starts analyzing the text based on the Keyword Extractor KER[3] [79] that uses TF-IDF to extract the most statistically significant mentions from the text as candidate classes in the vocabulary. This step does not involve any semantic technology since there is no semantic information present in the knowledge base yet. The extracted information from the text is then presented to the user as a highlighted text with actions. These actions allow the user to create a new term in the vocabulary. The user can reject the suggested term if it is irrelevant to the associated vocabulary.

- The second scenario has much in common with the previous one, but it suggests that the vocabulary already has seed classes. In addition to the steps introduced in the first scenario, Annotace starts analyzing the document using the classes in the associated vocabulary to find mentions in the text that refer to specific entities in the vocabulary and provides links between them. These mentions are also presented as highlighted text in the document, but differ from the extracted terms in the statistical step by providing a link to the associated term directly. Similarly to the create and reject actions, the user is allowed to approve the suggested association or change the association to a different term in the vocabulary.

Annotace performs a preprocessing step on the imported documents to augment the text with the syntactic information, and the lemmas for each token in the text as well as all terms' present labels in the vocabulary. It uses a morphological analyzer tool called MorphoDiTa, Morphological Dictionary and Tagger [80]. MorphoDiTa[4] uses trained language models for both the Czech and English languages. Other morphological analyzers can be easily plugged into Annotace, making the tool easily adapted to a wide variety of other languages.

Both scenarios suggest human interaction with the system to approve or reject the output of Annotace. The semi-automatic approach is paramount to keeping the high precision of building the ontology and saving the user time and effort needed to be spent with the manual process. Figure 6.2 shows the usage of Annotace within TermIt, where it is used to annotate the MPP document with terms from the MPP ontology.

Annotace handles data in HTML format and annotations are created using

[3] `https://github.com/ufal/ker` accessed: 2023-05-29

[4] `http://ufal.mff.cuni.cz/morphodita` accessed: 2023-05-29

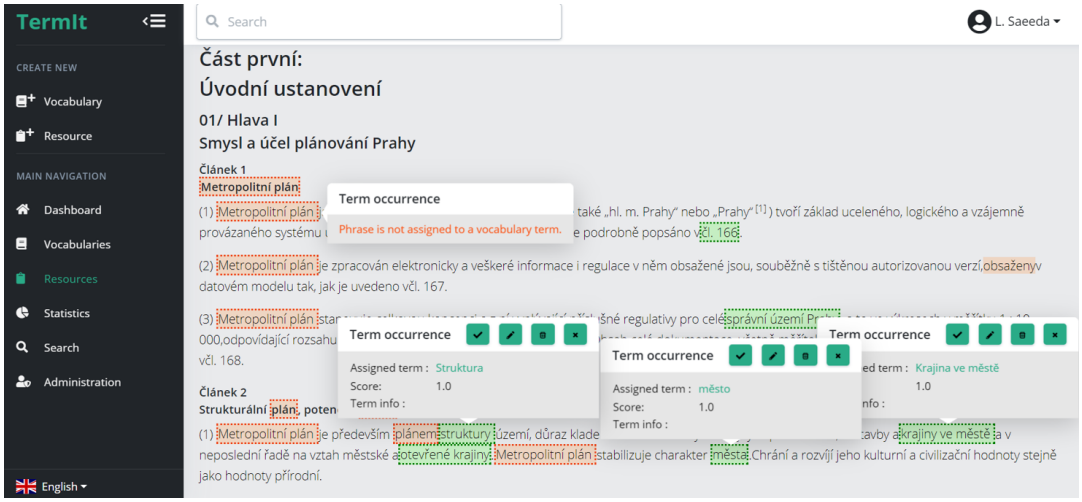


Figure 6.2: Annotace the text annotation service within TermIt

RDFa[5] [81]. RDFa is an extension to HTML5 that allows injecting linked data annotations into the structure of the HTML document. Whenever a token is recognized as an entity mention for an entity in the vocabulary, a new annotation is injected around this token with properties of this annotation, such as a unique ID, the resource attribute referring to the URI of the entity in the vocabulary, the type of annotation in the ontology model, and the accuracy of the prediction represented in the score attribute as depicted in Listing 1.

```
<html prefix="ddo:http://onto.fel.cvut.cz/ontologies/application/
    ↪ termit/pojem/">
        <p> Metropolitní plán vymezuje ve <span about="_:4"
            ↪ property="ddo:je-výskytem-termu" resource="http
            ↪ ://onto.fel.cvut.cz/ontologies/slovnik/datovy-
            ↪ mpp-3.5-np/pojem/správní-území-prahy" typeof="
            ↪ ddo:výskyt-termu" score="1.0">správním území
            ↪ Prahy</span> hranici zastavěného území... </p>
```

Listing 1: Annotated HTML with RDFa (output sample)

After annotating the documents by Annotace with the corresponding ontological classes, to incorporate the annotations created by Annotace in the LSPs, Annotace augments the output with their appropriate tags presented in Table 5.1 and parses the resulted files in an XML-based format that serves as input for the patterns implementation tool. Patterns were tested separately within the Hermes system to evaluate their efficiency. The results were discussed in further detail in Section 5.1. We consider integrating the patterns in the pipeline within TermIt as part of ongoing work.

[5]https://www.w3.org/TR/rdfa-core/

6.2 Reporting Tool

Reporting process of aviation safety incidents and accidents must be clearly and easily done. To achieve this, the Reporting Tool introduced in [82] has been built on top of the Aviation Safety Ontology (ASO). In order to make the reporting process more user-friendly, as well as make it easy and logical, a smart form generation based on the event type and other attributes is needed, in order to support the reporting process by reducing the list of attributes that have to be filled, only to those related and relevant for a specific event type. In order to detect the event type in the initial safety report, a comprehensive textual analysis process has to be performed, taking into consideration the unstructured nature of the initial input report, which is usually full of jargon.

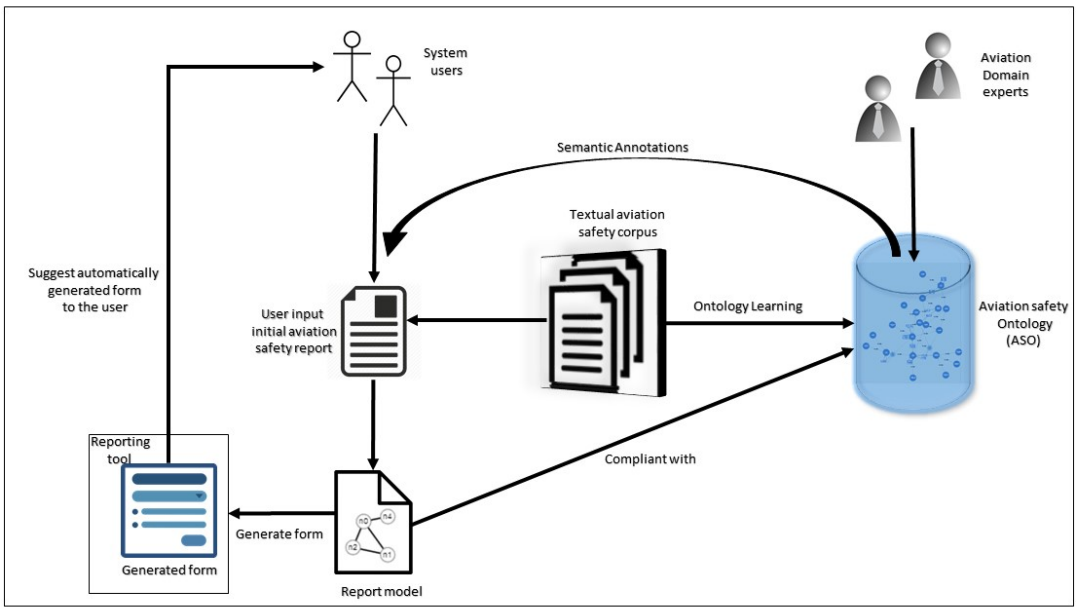


Figure 6.3: Full scenario of the iterative approach of annotating and learning back the ontology

The processing pipeline discussed in Section 4.2.1 focused on the OBIE task from the full pipeline depicted in Figure 6.3. It has achieved high-precision semantic annotations for aviation safety reports, detecting the main event and event-type in the safety report based on the ASO as well as all participants, temporal and spatial information, including all information that can help construct a dynamic form suited to the actual report.

An evolved ontology containing a class hierarchy relevant to the aviation safety domain can simplify the reporting process by enabling the reporter to process the data in a controlled way by means of an ontology. Therefore, the reporter will provide more relevant and accurate data. This will ensure a better experience for the safety management of the statistics Business Intelligent (BI) user who will benefit from the targeted, without noise, and less biased statistics,

which will improve the quality of the data, the speed of the reporting process on the general level, and provide more precise results.

The ontology learning part of the pipeline was not implemented as part of the tool but is considered for future system improvements and to contribute back to the ASO ontology.

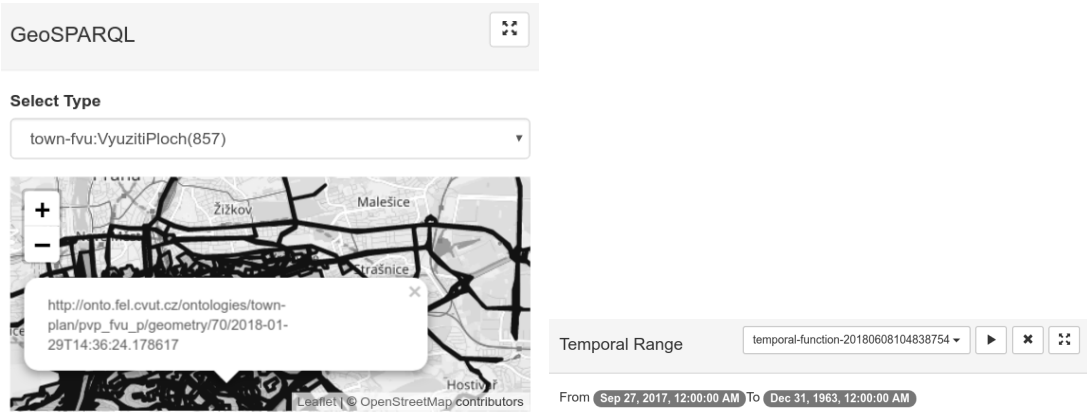
6.3 Reliability and Quality Knowledge System

This is a prototype system in the plan and design phase. The application of the methodology is only done partially as a proof-of-concept attempt. The scope of the Reliability and Quality Knowledge System (RQKS) system is the reliability and quality of aircraft systems and components. The system knowledge base is implemented using ontologies. The knowledge base is structured according to Reliability and Quality ontology which provides the main schema, and the Aviation Product Quality and Reliability (APQR) ontology contains concepts and relations relevant to the domain of discourse, i.e., aircraft systems and components in aviation The proposed GATE pipeline introduced in sections 4.2.2 is prototyped to be used in three use cases of the RQKS system. The first use case is to populate the instance data of the RQKS. The pipeline is used to find mentions and relations between mentions of concepts in the APQR ontology from the relevant documents. The results are converted to an ontology that can be merged with the instance ontology in the RQKS. The second use case is document indexing, which uses linked entities to generate a document index based on the annotated corpus. The combination of the entity linking and ontology learning pipeline presented in sections 4.2.2 and 5.2 is used to implement the ontology enrichment use case to learn new ontological entities in an iterative way.

6.4 Dataset Dashboard

The Dataset Dashboard[6] is a SPARQL endpoint exploration tool that helps to understand the structure of the dataset and the relationships to other datasets. The tool is based on the notion of a dataset descriptor that describes some characteristics of a dataset. The tool offers descriptors for basic class/ property statistics, spatial information, temporal information, as well as advanced dataset summarization [83]. The main purpose of the dataset dashboard is to compute and visualize descriptors of various types in the form of multiple widgets. For example, the Summary Schema Widget, the Spatial Widget, and the temporal widget. The temporal widget shows the temporal coverage of

[6]https://onto.fel.cvut.cz/dataset-dashboard last accessed 2023-05-29



(a) (b)

Figure 6.4: Example temporal/spatial descriptors of the dataset.

the dataset, computed as minimum and maximum time points occurring in the dataset. The computation itself considers the structured temporal information as well as temporal information extracted from non-structured texts. For the unstructured temporal information, it retrieves the textual literals and performs a natural language processing analysis to extract the time information. The pipeline to calculate the temporal descriptor can be found in Section 5.3.

As an example, consider a dataset g_{exp} about parcels, buildings, floors, and land use that is part of the dataset maintained by the Prague Institute of Planning and Development[7]. Compared to the original, g_{exp} is limited to the data for the Prague Center, which results in approximately 1.8 million triples. The spatial and temporal context of the dataset is shown in Figure 6.4. The temporal descriptor shows the temporal range of the dataset extracted using the pipeline described in Section 5.3. This dataset provides temporal information only inside the properties `town-parcely:dat_vznik` and `town-budovy:dat_vznik`, `town-budovy:dat_zmena`, which denote the creation/ change dates of the data change record. So in this case, the extracted temporal knowledge refers rather to the creation of the data than to the actual content. The GeoSPARQL descriptor is out of the scope of this work. For more information on the various available descriptors, refer to [83].

[7] `http://en.iprpraha.cz`, Accessed: 2023-05-29

Chapter 7

Conclusion

In this chapter of the thesis, we discuss the achieved results of our work, including the additional, still unanswered questions, and possibilities for extensions in future research.

7.1 Discussion

Our research was carried out through a series of projects that gave us partial results and evaluations.

However, we were able to apply the proposed method to real-world data where other existing techniques were difficult to apply.

Often, few languages are considered in ontology learning systems due to the unavailability of appropriate efficient primary tools/ resources [2].

The proposed entity linking method, although its simplicity, is extendible, where the performance can be further improved. The system can be easily adapted to other languages and serve as the starting point for developing the domain ontology for under-supported languages. It is also efficient for many other use cases, such as document indexing, searching, and text summarization.

Lexico-Semantic Patterns (LSPs) for ontology learning have several limitations. It can be brittle since it relies on a set of rules that may not be able to capture all the relevant information in the text. It can also be difficult to generalize since the rules may be specific to a particular domain or language.

On the other hand, LSPs work very well when applied to specific domain documents written in a language that lacks proper processing tools and has poorly annotated data. These methods are useful for more focused information extraction and ontology learning tasks where the output is well-designed. It can be relatively easy to implement and customize, even for non-experts. It can also be efficient and scalable since it can be applied to larger volumes of text automatically.

The entity linking system is essential for any ontology learning system to

add semantic context to the text. LSPs can be used in the early development stage of the ontology, but are generally effective with expert fine-tuning and well-curated vocabularies. For later stages, a combination of deep learning techniques can be used together with the rule-based ones.

for the iterative methodology of entity linking and ontology learning, it would be interesting to monitor the long-term effect of the repetitive process on the resulting ontology in terms of stabilizing the knowledge base with further iterations. In addition, measuring and evaluating the efficiency and time demands of involving the expert interaction with the system in different stages of the iterations to ensure the quality level. Then, check whether the human interaction will actually be reduced with more iterations.

7.2 Summary and Future Work

Ontology learning from unstructured text is challenging when applied to domains with low resources, such as the lack of processing tools that can handle the language of the text, the size of the corpus, or the availability of annotated data. This thesis introduced an end-to-end approach to extracting domain ontology from the text for such scenarios. The approach consists of a 1) *Knowledge Acquisition* phase where a seed ontology is built and used and input to the 2) *Entity Linking* phase to identify entity mentions in the available textual documents. Phase 3) *Ontology Learning* is where the Lexico-Semantic Patterns (LSPs) are used to learn new ontological entities. The output of each task in the learning pipeline is evaluated separately as a 4) *Quality Assurance* phase.

Phase 2, Entity Linking, and Phase 3, Ontology Learning are related tasks that can be used together, and hence, can be executed iteratively to for continuous ontology learning. The output of the entity linking can be used to help update the ontology, where the identified entities can be used in the LSPs to extract new concepts and establish the hierarchy of those concepts within the ontology. Conversely, the output of ontology learning can be used to improve entity linking by providing a more structured and comprehensive set of ontological entities that can be used to disambiguate mentions in the text.

For limited-resources languages that lack existing tools that can handle the ontology learning task, we presented a vanilla Entity Linking method that can be summarized by a *preprocessing* step, *candidate entity set generation*, and *candidate entity set scoring* steps. **Annotace**, the entity linking service was created to prototype the presented method on Czech resources, and by plugging an appropriate morphological analyzer, it can be easily adapted to other languages.

For the Ontology Learning phase, we described a rule-based relation extrac-

tion approach to support the ontology building process, based on a domain-specific seed vocabulary and textual documents. We defined a preliminary set of Lexico-Semantic Patterns corresponding to common ontological relations to help extract relations between concepts based on the analysis of annotated documents written in Czech.

For limited training data scenarios, we tested the pipeline using a combination of existing tools. First, we created an Entity Linking pipeline comprised of Apache Stanbol, Dbpedia Spotlight, and a set of simple extraction patterns customized for the domain studied. Flexible gazetteers and OntoRoot Gazetteers within the GATE framework were also utilized to implement the Entity Linking task.

GATE ontology-aware JAPE was used to implement Lexico-Semantic Patterns (LSPs) for extracting ontological entities from these texts and learning hierarchies. Finally, we provide an ontology learning enhancement pipeline to extract temporal knowledge from text and introduced the Temporal Descriptor Ontology that can be used to extend the domain ontology with the temporal dimension.

We showed some real-world applications and use cases where the proposed methodology was utilized either fully or partially, as in TermIt, the semantic vocabulary manager to annotate resources and create vocabularies based on documents, the Reporting Tool to process aviation safety incident and accident reports, the Reliability and Quality Knowledge System for updating the domain ontology and document indexing, and the Dataset Dashboard to describe datasets with their temporal data.

Annotace, the entity linking system is a prototype implementation that provided sufficient results. However, there is a window for improving the entity disambiguation, for example, by comparing the local term context (neighbor term mentions in the text) with its global context (neighbors of the term in the ontology) and incorporating the value in the scoring function.

Also, it can be useful to configure the preprocessing component of Annotace to support language models for other languages that are similar in nature to the Czech language.

To extend this work, it is possible to expand the introduced set of LSPs to cover more common ontological relationships. It is recommended to investigate the more flexible rule-based languages and tools available, taking into consideration the availability to plug the language-specific models. It is clear that a generic language-independent framework is needed that supports the development and processing of LSPs that comes with the application programming interface (API) that supports easy integration with other systems.

It can be useful to consider developing GATE plugins to support Czech

resources to make use of the GATE framework's various NLP capabilities.

Finally, since Annotace implementation also supports English resources, the pipeline described for languages with limited resources presented in Sections 4.1.2 and 5.1.1 can be applied to the aviation safety corpus presented in Section 4.2.1 and compare the performances of the two pipelines.

Appendix A

GATE Gazetteers

- Plugins needed: Tools, Ontology, Ontology_Based_Gazetteer, and ANNIE.
- Load the ontology into GATE.
- Create a new Corpus Pipeline named "Root finder", load and add the following processing resources to the pipeline.
  - ANNIE English Tokenizer (or any other GATE-compatible tokenizer)
  - ANNIE POS Tagger (or any other GATE-compatible POS tagger)
  - GATE Morphological Analyzer (or any other GATE-compatible morphological analyzer) This pipeline is necessary to pre-process the ontological entities and get the root of the terms.
- Create an OntoRoot Gazetteer processing resource, set the previously loaded ontology and the Root Finder application as input parameters. Other parameters can be adjusted as desired, including CaseSensitive, PropertiesToExclude, PropertiesToInclude, UseResourceUri, etc. The ontology will be pre-processed and analyzed in this Gazetteer initialization step.
- Create a Flexible Gazetteer and set the previously created OntoRoot Gazetteer as an input parameter. For inputFeatureNames parameter, click on the button on the right and add "Token.root" in the provided textbox, then click Add button. This allows to match against "root" feature of an annotation, not the whole string.
- Create a new Corpus Pipeline and add the following processing resources to it.
  - Document Reset processing resource
  - RegEx Sentence Splitter (or ANNIE Sentence Splitter)

- ANNIE English Tokenizer (or any other GATE-compatible tokenizer)
- ANNIE POS Tagger (or any other GATE-compatible POS tagger)
- GATE Morphological Analyzer (or any other GATE-compatible morphological analyzer)
- The previously created Flexible Gazetteer
- Create a document and process it with the created application.

Appendix B

General Notes on Creating JAPE Patterns

Macros are helpful when reusing the same patterns in several places in grammar. JAPE allows defining macros, which are labeled patterns that can be reused. For example, defining multiple strings under one value.

```
Macro: THE
(
 {Token.string == "the"}|
 {Token.string == "The"}|
 {Token.string == "THE"}
)
```

On the other hand, Templates are variables for a quoted string, a number or a boolean (true or false). String templates can have parameters and parameter values supplied in the call. It is useful if you have many similar strings in your grammar. For example:

```
Template: threshold = 0.6
Template: source = "Interesting location finder"
Rule: IsInteresting
({Location.score > [threshold]}):loc
-->
:loc.Entity = { kind = "Location", source = [source]}
```

Contextual Operators “contains” and “within” match annotations within the context of other annotations. For example, Organization contains Person matches if an Organization annotation completely contains a Person annotation. Person within Organization matches if a Person annotation lies completely within an Organization annotation. The difference between the two is that the first annotation specified is the one matched. In the first example, Organization is matched. In the second example, Person is matched.

Ontology Aware JAPE

OntoRoot Gazetteer puts class URIs in a feature called "classURI" and the instance URI in a feature called URI. Therefore, we need a JAPE grammar to go through each Lookup annotation and copy the value of its class URI feature to a "class" feature.

Example: find all lookup annotations produced by OntoRoot Gazetteer that are of type = instance takes the value of their classURI feature and copies it to the class features. Similarly, this happens for the instance URI, which is copied to the "inst" feature.

```
Phase: Lookup
RenameInput: Lookup
Options: control = applet
Rule: RenameLookup
({Lookup.type == instance}):match
->::match{
 For (Annotation lookup : matchAnnots) {
FeatureMap theFeatures = lookup.getFeatures();
theFeatures.put("class", theFeatures.get("classURI"));
theFeatures.put("inst", theFeatures.get("URI"));}}
```

Similar rule to create "class" feature to classes and subclasses of a specific type.

```
Phase: LookupRename
Input: Lookup
Options: control = appelt
Rule: RenameLookup (
        {Lookup.type == class}
):match
-->
:match{
       AnnotationSet theAnnots = bindings.get("match");
       if(theAnnots != null && theAnnots.size() != 0) {
       Annotation theLookup = theAnnots.iterator().next();
       FeatureMap theFeatures = theLookup.getFeatures();
       theFeatures.put("class", theFeatures.get("URI")); }}
```

Bibliography

[1] F. Gutierrez, D. Dou, S. Fickas, D. Wimalasuriya, and H. Zong, "A hybrid ontology-based information extraction system", *Journal of Information Science*, vol. 42, no. 6, pp. 798–820, 2016.

[2] A. C. Khadir, H. Aliane, and A. Guessoum, "Ontology learning: Grand tour and challenges", *Computer Science Review*, vol. 39, p. 100 339, 2021, ISSN: 1574-0137. DOI: https://doi.org/10.1016/j.cosrev.2020.100339. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1574013720304391.

[3] T. B. Brown, B. Mann, N. Ryder, *et al.*, *Language models are few-shot learners*, 2020. arXiv: 2005.14165 [cs.CL].

[4] J. H. Caufield, H. Hegde, V. Emonet, *et al.*, "Structured prompt interrogation and recursive extraction of semantics (spires): A method for populating knowledge bases using zero-shot learning", Apr. 2023.

[5] Z. Ji, N. Lee, R. Frieske, *et al.*, "Survey of hallucination in natural language generation", *ACM Comput. Surv.*, vol. 55, no. 12, 2023, ISSN: 0360-0300. DOI: 10.1145/3571730. [Online]. Available: https://doi.org/10.1145/3571730.

[6] T. R. Gruber, "A translation approach to portable ontology specifications", *Knowledge acquisition*, vol. 5, no. 2, pp. 199–220, 1993.

[7] P. N. Mendes, M. Jakob, and C. Bizer, "Dbpedia: A multilingual cross-domain knowledge base.", in *LREC*, 2012, pp. 1813–1817.

[8] C Masolo, S Borgo, A Gangemi, N Guarino, A Oltramari, and L Schneider, "Dolce: A descriptive ontology for linguistic and cognitive engineering", *WonderWeb Project, Deliverable D17 v2*, vol. 1, pp. 75–105, 2003.

[9] G. Guizzardi, *Ontological Foundations for Structural Conceptual Model*. 2005, vol. 015, p. 441, ISBN: 9075176813. DOI: 10.1007/978-3-642-31095-9_45. [Online]. Available: http://doc.utwente.nl/50826.

[10] G. Guizzardi, G. Wagner, R. de Almeida Falbo, R. S. S. Guizzardi, and J. P. A. Almeida, "Towards ontological foundations for the conceptual modeling of events", in *Conceptual Modeling*, W. Ng, V. C. Storey, and J. C. Trujillo, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 327–341, ISBN: 978-3-642-41924-9.

[11] A. Benevides, J.-R. Bourguet, G. Guizzardi, R. Peñaloza, and P. Pe˜naloza, "Representing the ufo-b foundational ontology of events in sroiq", Jan. 2017.

[12] J. R. Hobbs and F. Pan, "An Ontology of Time for the Semantic Web", *ACM Transactions on Asian Language Information Processing*, vol. 3, no. 1, pp. 66–85, 2004, ISSN: 15300226. DOI: 10.1145/1017068.1017073.

[13] J. F. Allen and G. Ferguson, "Actions and events in interval temporal logic", *Spatial and Temporal Reasoning*, 205–245, 1997. DOI: 10.1007/978-0-585-28322-7_7.

[14] P. Buitelaar, P. Cimiano, and B. Magnini, "Ontology learning from text: An overview", *Ontology learning from text: Methods, evaluation and applications*, vol. 123, pp. 3–12, 2005.

[15] W. Wong, W. Liu, and M. Bennamoun, "Ontology learning from text: A look back and into the future", *ACM Computing Surveys (CSUR)*, vol. 44, no. 4, p. 20, 2012.

[16] A. Mykowiecka, M. Marciniak, and A. Kupść, "Rule-based information extraction from patients' clinical data", *Journal of biomedical informatics*, vol. 42, no. 5, pp. 923–936, 2009.

[17] K. Nebhi, "Ontology-based information extraction from twitter", 2012.

[18] H. Cunningham, D. Maynard, K. Bontcheva, and V. Tablan, "A framework and graphical development environment for robust nlp tools and applications.", in *ACL*, 2002, pp. 168–175.

[19] S. Zitnik and M. Bajec, "Ontology-based information extraction: A machine learning approach",

[20] S. Soderland, "Learning information extraction rules for semi-structured and free text", *Machine learning*, vol. 34, no. 1, pp. 233–272, 1999.

[21] D. Dou, H. Wang, and H. Liu, "Semantic data mining: A survey of ontology-based approaches", in *Semantic Computing (ICSC), 2015 IEEE International Conference on*, IEEE, 2015, pp. 244–251.

[22] D. C. Wimalasuriya and D. Dou, *Ontology-based information extraction: An introduction and a survey of current approaches*, 2010.

[23] W. Shen, J. Wang, and J. Han, "Entity linking with a knowledge base: Issues, techniques, and solutions", *IEEE Transactions on Knowledge and Data Engineering*, 2015.

[24] S. Guo and et al., "To link or not to link? a study on end-to-end tweet entity linking", in *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics*, 2013.

[25] A. Gattani and et al., "Entity extraction, linking, classification, and tagging for social media: A wikipedia-based approach", *Proceedings of the VLDB Endowment*, 2013.

[26] W. Shen and et al., "Linking named entities in tweets with knowledge base via user interest modeling", in *Proceedings of the 19th ACM SIGKDD*, 2013.

[27] W. Zhang and et al., "Entity linking with effective acronym expansion, instance selection and topic modeling", in *22 International Joint Conference on AI*, 2011.

[28] J. Lehmann and et al., "Lcc approaches to knowledge base population at tac 2010.", in *TAC*, 2010.

[29] S. Gottipati and J. Jiang, "Linking entities to a knowledge base with query expansion", in *Proceedings of the Conference on Empirical Methods in NLP*, 2011.

[30] A. Jain, S. Cucerzan, and S. Azzam, "Acronym-expansion recognition and ranking on the web", in *2007 IEEE International Conference on Information Reuse and Integration*, 2007.

[31] W. Zhang and et al., "Nus-i2r: Learning a combined system for entity linking.", in *TAC*, 2010.

[32] L. Ratinov, D. Roth, D. Downey, and M. Anderson, "Local and global algorithms for disambiguation to wikipedia", in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, Association for Computational Linguistics, 2011.

[33] M. Konkol, "First steps in czech entity linking", in *International Conference on Text, Speech, and Dialogue*, Springer, 2015, pp. 489–496.

[34] A. Pilz and G. Paaß, "From names to entities using thematic context distance", in *Proceedings of the 20th ACM International Conference on Information and Knowledge Management*, 2011.

[35] Z. Chen and H. Ji, "Collaborative ranking: A case study on entity linking", in *Proceedings of the 2011 Conference on Empirical Methods in NLP*, 2011.

[36] D. M. Nemeskey, G. Recski, A. Zséder, and A. Kornai, "Budapestacad at tac 2010", in *TAC*, 2010.

[37] V. Varma and et al., "Iiit hyderabad in guided summarization and knowledge base population", 2019.

[38] X. Han and L. Sun, "A generative entity-mention model for linking entities with knowledge base", in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, 2011.

[39] S. Tirunagari, "Data mining of causal relations from text: Analysing maritime accident investigation reports", *arXiv preprint arXiv:1507.02447*, 2015.

[40] T. Williams, J. Betak, and B. Findley, "Text mining analysis of railroad accident investigation reports", in *2016 Joint Rail Conference*, American Society of Mechanical Engineers, 2016, V001T06A009–V001T06A009.

[41] M. Chong, A. Abraham, and M. Paprzycki, "Traffic accident analysis using machine learning paradigms", *Informatica*, vol. 29, no. 1, 2005.

[42] A. J.-P. Tixier, M. R. Hallowell, B. Rajagopalan, and D. Bowman, "Automated content analysis for construction safety: A natural language processing system to extract precursors and outcomes from unstructured injury reports", *Automation in Construction*, vol. 62, pp. 45–56, 2016.

[43] A. Chokor, H. Naganathan, W. K. Chong, and M. El Asmar, "Analyzing arizona osha injury reports using unsupervised machine learning", *Procedia engineering*, vol. 145, pp. 1588–1593, 2016.

[44] K. Fujita, M. Akiyama, K. Park, E. N. Yamaguchi, and H. Furukawa, "Linguistic analysis of large-scale medical incident reports for patient safety.", in *MIE*, 2012, pp. 250–254.

[45] C. H. Hwang, "Incompletely and imprecisely speaking: Using dynamic ontologies for representing and retrieving information.", in *KRDB*, vol. 21, 1999, pp. 14–20.

[46] M. Baroni and S. Bernardini, "Bootcat: Bootstrapping corpora and terms from the web.", in *LREC*, 2004, p. 1313.

[47] R. Sombatsrisomboon, Y. Matsuo, and M. Ishizuka, "Acquisition of hypernyms and hyponyms from the www", in *Proceedings of the 2nd International Workshop on Active Mining*, 2003.

[48] M. A. Hearst, "Automated discovery of wordnet relations", *WordNet: an electronic lexical database*, pp. 131–153, 1998.

[49] K. Lindén, J. O. Piitulainen, *et al.*, "Discovering synonyms and other related words", in *Proceedings of COLING 2004 CompuTerm 2004: 3rd International Workshop on Computational Terminology*, 2004.

[50] H. N. Fotzo and P. Gallinari, "Learning «generalization/specialization» relations between concepts: Application for automatically building thematic document hierarchies", in *Coupling approaches, coupling media and coupling languages for information retrieval*, 2004, pp. 143–155.

[51] T. Jiang, A.-H. Tan, and K. Wang, "Mining generalized associations of semantic relations from textual web content", *IEEE transactions on knowledge and data engineering*, vol. 19, no. 2, pp. 164–179, 2007.

[52] A. Rios-Alvarado and I. Lopez-Arevalo, "Ontology learning from text: Method for learning axioms", Technical report, Tech. Rep., 2012.

[53] A. B. Rios-Alvarado, I. Lopez-Arevalo, E. Tello-Leal, and V. J. Sosa-Sosa, "An approach for learning expressive ontologies in medical domain", *Journal of medical systems*, vol. 39, no. 8, p. 75, 2015.

[54] P. Cimiano and J. Volker, "Text2onto - a framework for ontology learning and data-driven change discovery", *Lecture Notes in Computer Science*, 2005.

[55] A. Gangemi, V. Presutti, D. R. Recupero, A. G. Nuzzolese, F. Draicchio, and M. Mongiovi, "Semantic Web Machine Reading with FRED", *Semantic Web*, vol. 8, no. 6, pp. 873–893, 2017.

[56] H. Cunningham, D. Maynard, and V. Tablan, "Jape: A java annotation patterns engine", 1999.

[57] H. Cunningham, "Gate, a general architecture for text engineering", *Computers and the Humanities*, 2002. [Online]. Available: `https://doi.org/10.1023/A:1014348124664`.

[58] D. Maynard, A. Funk, and W. Peters, "Using lexico-syntactic ontology design patterns for ontology creation and population", in *Proceedings of the 2009 International Conference on Ontology Patterns-Volume 516*, CEUR-WS. org, 2009, pp. 39–52.

[59] W. Ijntema, J. Sangers, F. Hogenboom, and F. Frasincar, "A lexico-semantic pattern language for learning ontology instances from text", *Web Semant.*, vol. 15, pp. 37–50, Sep. 2012, ISSN: 1570-8268. DOI: `10.1016/j.websem.2012.01.002`. [Online]. Available: `http://dx.doi.org/10.1016/j.websem.2012.01.002`.

[60] J. Borsje, F. Hogenboom, and F. Frasincar, "Semi-automatic financial events discovery based on lexico-semantic patterns", *International Journal of Web Engineering and Technology*, vol. 6, no. 2, p. 115, 2010.

[61] M. Hamroun and et al., "Lexico semantic patterns for customer intentions analysis of microblogging", in *2015 11th International Conference on Semantics, Knowledge and Grids (SKG)*, 2015.

[62] A. K. Kolya, A. Ekbal, and S. Bandyopadhyay, "A Supervised Machine Learning Approach for Temporal Information Extraction 3 Conditional Random Field Based Approach", pp. 447–454, 2007.

[63] B. Tang, Y. Wu, M. Jiang, Y. Chen, J. C. Denny, and H. Xu, "A hybrid system for temporal information extraction from clinical text.", *Journal of the American Medical Informatics Association : JAMIA*, vol. 20, no. 5, pp. 828–35, 2013, ISSN: 1527-974X. DOI: 10.1136/amiajnl-2013-001635. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/23571849.

[64] Y. K. Lin, H. Chen, and R. A. Brown, "MedTime: A temporal information extraction system for clinical narratives", *Journal of Biomedical Informatics*, 2013, ISSN: 15320464. DOI: 10.1016/j.jbi.2013.07.012.

[65] J. Strötgen and M. Gertz, "HeidelTime: High quality rule-based extraction and normalization of temporal expressions", *Proceedings of the 5th International Workshop on Semantic Evaluation*, no. July, pp. 321–324, 2010. [Online]. Available: http://dl.acm.org/citation.cfm?id=1859735.

[66] A. X. Chang and C. D. Manning, "SUTime: A library for recognizing and normalizing time expressions.", *Lrec*, no. iii, pp. 3735–3740, 2012, ISSN: 1098-6596. DOI: 10.1017/CBO9781107415324.004. arXiv: arXiv:1011.1669v3. [Online]. Available: http://www-nlp.stanford.edu/pubs/lrec2012-sutime.pdf.

[67] N. UzZaman, H. Llorens, L. Derczynski, M. Verhagen, J. Allen, and J. Pustejovsky, "Semeval-2013 task 1: {T}empeval-3: Evaluating time expressions, events, and temporal relations", *Second joint conference on lexical and computational semantics (* SEM)*, vol. 2, no. SemEval, pp. 1–9, 2013. eprint: 1206.5333.

[68] A. Magueresse, V. Carles, and E. Heetderks, *Low-resource languages: A review of past work and future challenges*, 2020. arXiv: 2006.07264 [cs.CL].

[69] https://github.com/Alir3z4/stop-words/blob/master/czech.txt, Accessed: 2019-04-11.

[70] B. Kostov, J. Ahmad, and P. Křemen, "Towards ontology-based safety information management in the aviation industry", in *OTM Confederated International Conferences" On the Move to Meaningful Internet Systems"*, Springer, 2016, pp. 242–251.

[73] R. Falbo, "Sabio: Systematic approach for building ontologies", *CEUR Workshop Proceedings*, vol. 1301, Jan. 2014.

[74] G. A. De Cea and et al., "Natural language-based approach for helping in the reuse of ontology design patterns", in *International Conference on Knowledge Engineering and Knowledge Management*, Springer, 2008, pp. 32–47.

[75] World Wide Web Consortium, "Data Catalog Vocabulary (DCAT)", *W3C*, no. January, 2014. [Online]. Available: http://www.w3.org/TR/vocab-dcat/$\backslash$nhttp://www.w3.org/TR/2014/REC-vocab-dcat-20140116/$\backslash$nhttp://www.w3.org/ns/dcat.rdf.

[76] M. Blaško, B. Kostov, and P. Křemen, "Ontology-based Dataset Exploration – A Temporal Ontology Use-Case", in *INTELLIGENT EXPLORATION OF SEMANTIC DATA (IESD 2016)*, Kode, 2016.

[77] F. B. Ruy, R. de Almeida Falbo, M. P. Barcellos, and G. Guizzardi, "An ontological analysis of the iso/iec 24744 metamodel.", in *FOIS*, 2014, pp. 330–343.

[79] J. Libovický, *KER - keyword extractor*, LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University, 2016.

[80] J. Straková, M. Straka, and J. Hajič, "Open-source tools for morphology, lemmatization, pos tagging and named entity recognition", in *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, 2014.

[81] B Adida, M Birbeck, S McCarron, and I Herman, "Rdfa core 1.1", *W3C technical reports*, 2010.

[82] P. Vittek, A. Lališ, S. Stojić, and V. Plos, "Challenges of implementation and practical deployment of aviation safety knowledge management software", in *International Conference on Knowledge Engineering and the Semantic Web*, Springer, 2016, pp. 316–327.

List of candidate's work related to the thesis

The percentage is even for all listed authors at each publication unless otherwise specified.

Journals (Under Review)

- P. Kremen, M. Med, M. Blaško, *et al.*, "Termit: Managing legal thesauri", *Semantic Web – Interoperability, Usability, Applicability an IOS Press Journal*,

Conferences

- L. Saeeda (60% contribution), M. Med, M. Ledvinka, *et al.*, "Entity linking and lexico-semantic patterns for ontology learning", in *The Semantic Web*, A. Harth, S. Kirrane, A.-C. Ngonga Ngomo, *et al.*, Eds., Cham: Springer International Publishing, 2020, pp. 138–153, ISBN: 978-3-030-49461-2

  The paper has been cited in:

  - B. Abbasi, I. Fatima, H. Mukhtar, *et al.*, "Autonomous schema markups based on intelligent computing for search engine optimization", *PeerJ Computer Science*, vol. 8, e1163, Dec. 2022. DOI: 10.7717/peerj-cs.1163
  - K. Ovchinnikova, I. Kononenko, and E. Sidorova, "Development of lexico-syntactic ontology design patterns for information extraction of scientific data", English, *CEUR Workshop Proceedings*, vol. 3036, pp. 349–361, 2021, Supplementary 23rd International Conference on Data Analytics and Management in Data Intensive Domains, DAMDID/RCDL 2021 ; Conference date: 26-10-2021 Through 29-10-2021, ISSN: 1613-0073
  - N. Shroff, P. Vandenbussche, V. Moore, *et al.*, "Supporting ontology maintenance with contextual word embeddings and maximum mean discrepancy", in *Joint Proceedings of the 2nd International Workshop on Deep Learning meets Ontologies and Natural Language Processing*, ser. CEUR Workshop Proceedings, vol. 2918, CEUR-WS.org, 2021, pp. 11–19. [Online]. Available: http://ceur-ws.org/Vol-2918/paper2.pdf
  - M. A. Stranisci, V. Patti, and R. Damiano, "Representing the underrepresented: A dataset of post-colonial, and migrant writers", in *3rd Conference on Language, Data and Knowledge, LDK 2021*, Schloss Dagstuhl-Leibniz-Zentrum fur Informatik GmbH, Dagstuhl Publishing, vol. 93, 2021, pp. 1–14
  - M. A. Stranisci, V. Basile, R. Damiano, *et al.*, "Mapping biographical events to odps through lexico-semantic patterns?", in *12th Workshop on Ontology Design and Patterns, WOP 2021*, CEUR-WS, vol. 3011, 2021, pp. 1–12

- L. Saeeda and P. Křemen, "Temporal knowledge extraction for dataset discovery", in *Proceedings of the 4th International Workshop on Dataset*

PROFIling and fEderated Search for Web Data (PROFILES 2017). Co-located with The 16th International Semantic Web Conference (ISWC 2017), CEUR Vol-1927

The paper has been cited in:

- P. Kremen, L. Saeeda, M. Blasko, *et al.*, "Dataset dashboard - a SPARQL endpoint explorer", in *Proceedings of the Fourth International Workshop on Visualization and Interaction for Ontologies and Linked Data co-located with the 17th International Semantic Web Conference, VOILA@ISWC 2018, Monterey, CA, USA, October 8, 2018*, ser. CEUR Workshop Proceedings, vol. 2187, CEUR-WS.org, 2018, pp. 70–77. [Online]. Available: http://ceur-ws.org/Vol-2187/paper7.pdf
- L. Lu, J. Fang, P. Zhao, *et al.*, "Eliminating temporal conflicts in uncertain temporal knowledge graphs", in *Web Information Systems Engineering – WISE 2018*, Cham: Springer International Publishing, 2018, pp. 333–347

- L. Saeeda, "Iterative approach for information extraction and ontology learning from textual aviation safety reports", in *The Semantic Web: 14th International Conference, ESWC 2017, Portorož, Slovenia, May 28 – June 1, 2017, Proceedings, Part II*, E. Blomqvist, D. Maynard, A. Gangemi, *et al.*, Eds. Cham: Springer International Publishing, 2017, pp. 236–245, ISBN: 978-3-319-58451-5. DOI: 10.1007/978-3-319-58451-5_18. [Online]. Available: https://doi.org/10.1007/978-3-319-58451-5_18

  The paper has been cited in:

  - M. Ledvinka and P. Kremen, "A comparison of object-triple mapping libraries", *Semantic Web journal*, vol. 11, pp. 483–524, Apr. 2020. DOI: 10.3233/SW-190345
  - P. Hughes, R. Robinson, M. Figueres-Esteban, *et al.*, "Extracting safety information from multi-lingual accident reports using an ontology-based approach", *Safety Science*, vol. 118, pp. 288–297, 2019, ISSN: 0925-7535. DOI: https://doi.org/10.1016/j.ssci.2019.05.029. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0925753518307586
  - U. Shoaib, L. Fiaz, C. Chakraborty, *et al.*, "Context aware urdu information retrieval system", *ACM Trans. Asian Low-Resour. Lang. Inf. Process.*, 2022, ISSN: 2375-4699. DOI: 10.1145/3502854. [Online]. Available: https://doi.org/10.1145/3502854
  - I. Grossoni, P. Hughes, Y. Bezin, *et al.*, "Observed failures at railway turnouts: Failure analysis, possible causes and links to current and future research", *Engineering Failure Analysis*, vol. 119, p. 104 987, 2021, ISSN: 1350-6307. DOI: https://doi.org/10.1016/j.engfailanal.2020.104987. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1350630720315119

- L. Saeeda, P. Křemen, and M. Štumper, "Text analyzing of aviation safety reports", in *11th Workshop on Intelligent and Knowledge Oriented Technologies 35th Conference on Data and Knowledge 2016*, ISBN: 978-80-227-4619-9

- P. Kremen, L. Saeeda, M. Blasko, *et al.*, "Dataset dashboard - a SPARQL endpoint explorer", in *Proceedings of the Fourth International Workshop on Visualization and Interaction for Ontologies and Linked Data co-located with the 17th International Semantic Web Conference, VOILA@ISWC 2018, Monterey, CA, USA, October 8, 2018*, ser. CEUR Workshop Proceedings, vol. 2187, CEUR-WS.org, 2018, pp. 70–77. [Online]. Available: `http://ceur-ws.org/Vol-2187/paper7.pdf`

  The paper has been cited in:

  - M. Ledvinka, M. Blaško, and P. Křemen, "Factors of efficient semantic web application development", in *On the Move to Meaningful Internet Systems. OTM 2018 Conferences*, Cham: Springer International Publishing, 2018, pp. 565–572
  - P. Grimmel, J. Wessel, M. Mennenga, *et al.*, "Potentials of ontology-based knowledge discovery in data bases for learning factories", *Available at SSRN 4073026*, 2022
  - P. McBrien and A. Poulovassilis, "A conceptual modelling approach to visualising linked data", in *On the Move to Meaningful Internet Systems: OTM 2019 Conferences*, Cham: Springer International Publishing, 2019, pp. 227–245, ISBN: 978-3-030-33246-4
  - A. Menin, C. Faron, O. Corby, *et al.*, "From Linked Data Querying to Visual Search: Towards a Visualization Pipeline for LOD Exploration", in *WEBIST 2021 - 17th International Conference on Web Information Systems and Technologies*, ser. Proceedings of the 17th International Conference on Web Information Systems and Technologies (WEBIST), Online Streaming, France, Oct. 2021. DOI: `10.5220/0010654600003058`. [Online]. Available: `https://hal.science/hal-03404572`
  - A. Menin, M. N. Do, C. Dal Sasso Freitas, *et al.*, "Using chained views and follow-up queries to assist the visual exploration of the web of big linked data", *International Journal of Human–Computer Interaction*, pp. 1–17, 2022
  - P. Maillot, O. Corby, C. Faron, *et al.*, "Indegx: A model and a framework for indexing rdf knowledge graphs with sparql-based test suits", *Journal of Web Semantics*, vol. 76, p. 100 775, 2023, ISSN: 1570-8268. DOI: `https://doi.org/10.1016/j.websem.2023.100775`. [Online]. Available: `https://www.sciencedirect.com/science/article/pii/S1570826823000045`
  - A. Menin, P. Maillot, C. Faron, *et al.*, "Ldviz: A tool to assist the multidimensional exploration of sparql endpoints", in *Web Information Systems and Technologies*, M. Marchiori, F. J. Domínguez Mayo, and J. Filipe, Eds., Cham: Springer International Publishing, 2023, pp. 149–173

- M. Ledvinka, P. Kremen, L. Saeeda (30% contribution), *et al.*, "Termit: A practical semantic vocabulary manager", in *Proceedings of the 22nd International Conference on Enterprise InformationSystems, ICEIS 2020, Prague, Czech Republic, May 5-7, 2020, Volume 1*, J. Filipe, M. Smialek, A. Brodsky, *et al.*, Eds., SCITEPRESS, 2020, pp. 759–766. DOI: `10.`

5220/0009563707590766. [Online]. Available: https://doi.org/10.5220/0009563707590766

The paper has been cited in:

– M. Kála, A. Lališ, and T. Vojtěch, "Analyzing aircraft maintenance findings with natural language processing", *Transportation Research Procedia*, vol. 65, pp. 238–245, 2022, 11th International Conference on Air Transport – INAIR 2022, Returning to the Skies, ISSN: 2352-1465. DOI: https://doi.org/10.1016/j.trpro.2022.11.028. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2352146522006950