Using Machine Learning for Ontology Engineering on EU Vocabularies

George Patrikios

SID: 3308180017

SCHOOL OF SCIENCE & TECHNOLOGY

A thesis submitted for the degree of

Master of Science (MSc) in Data Science

JANUARY 2021

THESSALONIKI – GREECE

Using Machine Learning for Ontology Engineering on EU Vocabularies

George Patrikios

SID: 3308180017

Supervisor: Dr Christos Berberidis

SCHOOL OF SCIENCE & TECHNOLOGY

A thesis submitted for the degree of

Master of Science (MSc) in Data Science

JANUARY 2021

THESSALONIKI – GREECE

Abstract

In our days, the Semantic Web has gained a lot of popularity since it provides standard ways regarding sharing and retrieving data. A key component of the Semantic Web is Ontology Engineering where it includes the tasks of ontology development and ontology alignment. These kinds of tasks require extensive human labor and a profound domain knowledge. There is a great need of automated solutions in Ontology Engineering. Machine Learning techniques are applied to various domains in order to provide experts with such solutions.

This thesis investigates the application of Machine Learning in Ontology Engineering by applying such techniques in the domain of e-Government and particularly on European Union's Vocabularies. Specifically, this thesis has two objectives: a) Solve the problem of "Sub-property Link Prediction" in an ontology set. b) Introduce an "Ontology Search Tool" based on pre-trained vector representations (text-embeddings). The experimental results are inspiring and indicate that Machine Learning techniques are applicable in Ontology Engineering.

Acknowledgements

I wish to thank my supervisor Dr. Christos Berberidis for his guidance and assistance throughout my studies. Also, I would like to thank the professors, all of whom I had the chance to meet and elaborate during my studies. In addition, I would particularly like to thank the PhD candidate, Konstantinidis Ioannis, for providing me with his valuable feedback, experience and time during this thesis.

Furthermore, I would like to express my deepest gratitude to Angela and my dear friends, Giannis and Vaggelis. Finally, I devote this thesis to my family, Anastasios-Aphrodite-Despoina, for their unconditional love and support throughout my years of study.

George Patrikios

04/01/2021

Contents

1 Introduction

Over the recent years, the amount of data available on the Web has grown exponentially. There is a great need for a common framework that allows the exploitation and reuse of these data. Semantic Web (SW) provides such a solution by enabling users to find, share and combine data[1]. A key advantage of the SW is that data can be interpreted by machines [1]. SW has proven over the years that is applicable in industries like biology and human sciences and has a great potential regarding other industries [2]. There are also numerous studies-that aim to incorporate SW with e-Government [3]. Data on SW are defined using agreed ontologies. There are many definitions of an ontology throughout the literature. A definition that stands out suggests the following [4]:

"an ontology specifies a rich description of the:

- terminology, concepts, nomenclature
- relationships among and between concepts and individuals
- sentences distinguishing concepts, refining definitions and relationships (constraints, restrictions, regular expressions)

relevant to a particular domain or area of interest" (Kendall & McGuinness, 2019, p. 2).

Ontologies enable data to be represented in a common structured manner based on RDF, RDFS and OWL languages while using commonly agreed vocabularies from the Semantic Web community. An ontology consists of two aspects: a) the vocabularies that define the naming of classes, properties and relationships and b) the controlled vocabularies that define the common lists of possible property values (e.g., an agreed list of country names).

Ontologies are also widely explored by European Union's authorities for cross-border data exchange between different Member States. Towards achieving cross-border semantic interoperability, the European Commission in cooperation with the World Wide Web Consortium (W3C) has published ontologies on different domains, like e-Government. W3C is an international community that aims at developing Web standards. The EU Core

[1] https://www.w3.org/2001/sw/Activity

Vocabularies along with the DCAT Application Profile for Data Portals in Europe and the Asset Description Metadata Schema (ADMS) consist of some of the great outcomes of such initiatives. The creation and promotion of data standards is crucial for Semantic Interoperability. Ontologies rely on experts' subjectivity meaning that different Ontologies may be created in a certain domain of knowledge. Due to this fact, ontology alignment techniques are necessary in order to achieve knowledge integration and thus semantic interoperability.

Ontologies are created by domain experts throughout a process named ontology engineering (OE). OE includes a set of procedures regarding processes of the ontology development, the ontology alignment, the ontology life cycle, tools, methodologies and languages [5]. However, ontology engineering requires extensive manual labor for the identification of the proper ontologies that describe the concepts of interest as there is an enormous amount of published data standards. As an initial step towards solving this problem, the research community has created the Linked Open Vocabularies (LOV), which is a catalogue of published vocabularies and ontologies [6]. LOV also includes a tool for searching on the catalogue that helps ontology engineers easily identify existing vocabularies related to their concepts of interest. However, many vocabularies that are included in LOV are out of date and are not W3C recommendations, which decreases the quality of the search results of the tool. Another initiative from the Semantic Interoperability Community (SEMIC) is the Core Data Model Mapping Directory that hosts a collection of mappings between the Core Vocabularies and related Core Data Models and includes search capabilities. However, this tool applies only to the EU Core Vocabularies. Another important aspect of ontology engineering is ontology alignment, where the task is to identify similar entities with an exact or narrow match to be able to merge various data on the Semantic Web. This process also requires extensive human labor. Therefore, there is a need to develop automated solutions that will help ontology engineers with the tasks of ontology development and ontology alignment.

The objectives of this thesis are concerning the improvement of ontology engineering by implementing machine learning techniques. There are two main objectives: a) Firstly, the problem of ‘Link prediction in a set of ontologies’ is encountered. More thoroughly, the ontologies used are concerning the domain of e-Government in European Union. By implementing machine learning techniques along with pre-trained vector representation models, this thesis focuses on capturing the existence of relation among properties in a

set of EU's ontologies. b) Secondly, an 'Ontology search tool' based on pre-trained vector representations is introduced aiming to help experts with the tasks of ontology development and ontology alignment. Specifically, the purpose of this tool is to help ontology engineers identify the entities and ontologies of their interest by providing a search query.

The rest of the thesis is organized as follows: Chapter 2 sets a background of word vector representations together with ontology vector representations. Moreover, it provides an overview of the ontologies retrieved from the EU's Publications Office. Chapter 3 gives an analysis of the dataset generated for the experiments as well as a description of the pre-processing pipeline, the pre-trained vector representation models and the machine learning algorithms implemented. In Chapter 4 the comparative results are demonstrated regarding the different techniques applied. Chapter 5 discusses the finding of this thesis and detects some limitations. Finally, chapter 6 gives a conclusion of what has been achieved and also points out a future direction.

2 Literature Review

This chapter provides a review of the research on literature on text vector representations (text embeddings) and ontology vector representations based on text embeddings. Furthermore, it provides an overview of the ontologies retrieved from the European Union's Publications Office.

2.1 Text embeddings

Word embeddings are vector representations of words which enable numerous operations between text and machines. This allows machine learning algorithms to understand the meaning behind words. In addition, word embeddings can be observed as a set of techniques which aim to represent words as real valued-vectors in a vector space. Words are represented by unique vector representations and similar words must have similar vector representations. One of the main advantages of such approaches is that these vectors are dense and have less features in comparison to one-hot encoding vectors. Having these characteristics, they could be used by neural networks in various machine learning tasks [7]. These distributed representations are generated based on the usage of words in a corpus. This means that words used in similar contexts will be represented by similar word embeddings. This is based on the "distributional hypothesis" where it is assumed that words with similar context should have similar meaning [8].

There are two main categories of word embeddings techniques: Frequency based Embedding and Prediction based Embedding [9]. In the first category there are three types of vector models: Count Vector, TF-IDF Vector and Co-Occurrence Vector.

Count Vector generates the embedding of each word based on a vocabulary which is created according to a corpus of documents [10]. The number of times a word appears in a document is counted resulting in a "word to document" matrix. The size of such matrix is [D X T] where D is the number of documents and T the number of terms in the vocabulary.

The second frequency based embedding model are TF-IDF vectors [11]. This model uses term-frequency and inverse document-frequency to create a “word to document” matrix. Specifically, this method considers the occurrence of words in the entire corpus providing a more reliable score for infrequent but rather significant terms.

Finally, Co-Occurrence vectors are produced based on the Co-Occurrence matrix. In order to create this matrix a fixed context window must be declared. Then the co-occurrence of words is computed by counting how many times they occur inside a predefined context window. After that, the matrix generated is decomposed using principal component analysis and singular value decomposition techniques generating the word embeddings [12].

On the other hand, Prediction based Embeddings like Word2vec [13] and Glove [14] tend to predict a word in a given context by assigning probabilities to words used for analogy and similarity tasks.

One of the most widely used word embedding methods is Word2vec. It is an unsupervised technique using a two-layer neural network in order to generate the word embeddings from contextual information. As input Word2vec accepts a corpus of words produced by a number of documents. The aim of Word2vec is to project every word of this corpus in the vector space.

Its word is represented by a unique embedding and similar words must have similar embeddings. Regarding Word2vec’s architecture, two main models exist: CBOW [15] and Skip-Gram [13].

The CBOW model generates a vector representation from a word by predicting this word based on its surrounding words. In detail, the neural network model accepts as input the surrounding words inside a fixed-sliding window and tries to predict the target word. The basic case of CBOW is setting the parameter of the fixed-sliding window equal to 1 word as demonstrated in figure. Considering a corpus size of V both the input and output layers are one-hot encoded of size [1 X V] except the cell that indexes the position of the target word. According to its architecture two sets of weights exist. The first one is located between the input and the hidden layer while the second one is between the hidden and the output layer and their sizes are [V X N] and [N X V] respectively. The hyper-parameter N defines the number of neurons in the hidden layer which is also the number of the dimensions of the vector representation.

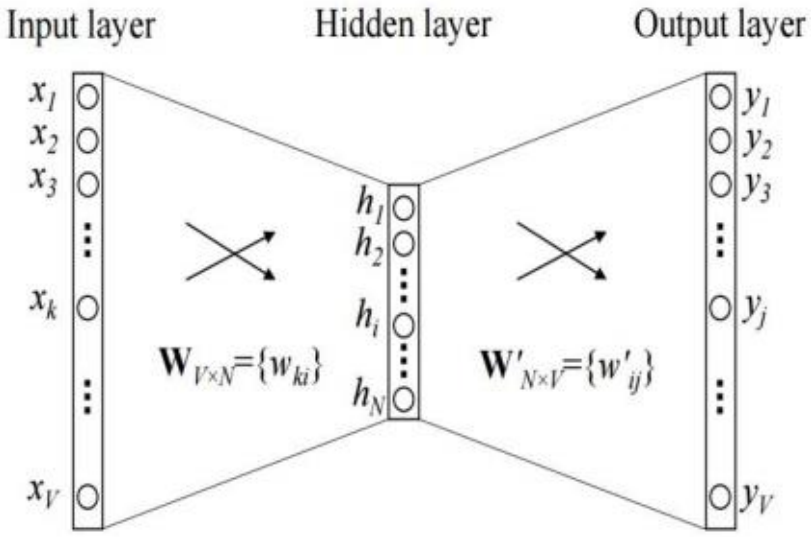


Figure 1: A CBOW model with only one word in the context

On the other hand, Skip-gram's topology can be considered as an inverse CBOW. In this model the neural network aims to predict the surrounding words from a current word. As in CBOW, a fixed-sliding window must be set. The model accepts as input a word and tries to predict the surrounding word inside the fixed-sliding window. The figures illustrate the topology of both models CBOW and Skip-Gram.

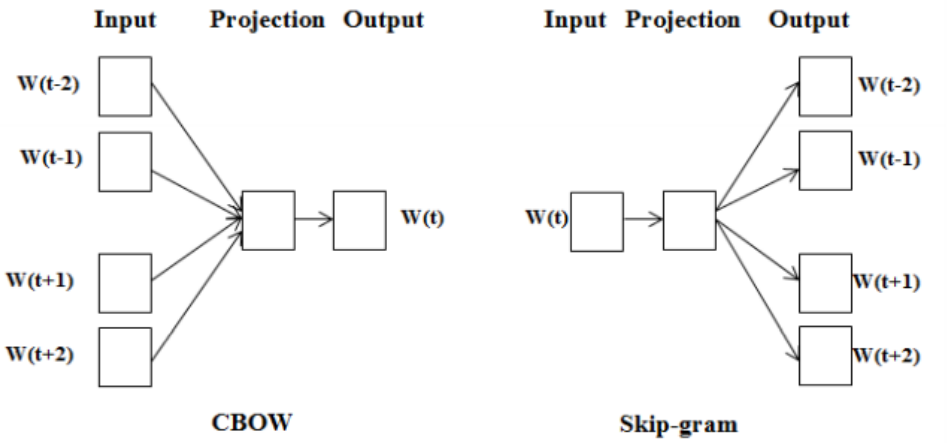


Figure 2: The CBOW model predicts the current word based on the context. The Skip-gram model predicts surrounding words given the current word

Global Vectors (GloVe) is an extension of the Word2vec method for generating word embeddings. Approaches like Latent Semantic Analysis (LSA) are efficient in learning global text statistics rather than local statistics like Word2vec. GloVe algorithm is based on both approaches by combining global and local statistics in order to generate the word vector representations. This means that GloVe combines both the local context information of words along with their word-occurrences in a corpus.

Apart from Word2vec and Glove, FastText is another word embedding approach [16]. It is an extension of Word2vec with the only difference that it feeds n-grams into a Neural Network instead of individual words. The vector representation for words, are produced by adding up the vector representations of their n-grams.

ELMo word embeddings are generated using a two-layer bidirectional language model [17]. This model accepts as an input word characters rather an entire word. Moreover, the vector representation produced for a word is actually a function of the entire sentence containing that word. This is a crucial characteristic of this method since it tackles the phenomenon of polysemy.

Much like ELMo, BERT tackles polysemy by generating contextualized embeddings which may differ according to the context of the sentence [18]. It is a deeply bidirectional, unsupervised language representation model which is pre-trained on a corpus. Specifically, the contextual relations among words are learned by utilizing a Transformer. The

Transformer reads the entire sequence of words in one pass allowing the model to learn the context of a word based on its surrounding words.

Another thing worth mentioning is the existence of pre-trained word embeddings. These kinds of embeddings are considered to capture general language aspects since they are generated by training in a large collection of documents consisting of billions of words. Then it can be used in order to solve other problems. Techniques like this enable Transfer Learning [19] which offers various benefits like improving the overall-performance, avoiding over-modelling and demanding less computational resources in machine learning tasks.

2.2 Text Embeddings for Ontology Engineering

Machine learning algorithms demand a set of features that are informative and discriminating. An ontology is specified as a set of triplets (C, P, I) where 'C' stands for the classes (subject) which represent the concept. In addition, 'P' stands for the properties denoting the relations (predicate) while 'I' indicates the individuals (object) [20].This specification allows ontologies to be interpreted as a graph network. Applying machine learning tasks to graph networks creates the need to construct a feature vector representation for the nodes and edges. To do that, the knowledge of those networks, both in the instance and ontology level, should be transformed into numerical representations aka embeddings. Producing high quality embeddings regarding ontology concepts is crucial for machine learning oriented ontology alignment, induction and enrichment tasks.

Generating ontology vector representations can be accomplished by employing word embeddings. In their study, Y. Zhang and X. Wang [21] aim to generate ontology embeddings by using word embeddings. It is worth mentioning that it is one of the first studies which tries to merge word embeddings with ontologies. To do so, a hybrid method is used in order to combine word embeddings and edit distance. The word embeddings are generated using Wikipedia. Edit distance is a string based metric which measures the similarity among two words and is used in various matching systems like RiMOM [22], ASMOV [23] and AgreementMaker [24]. In detail, an element-level matcher is introduced

which accepts two ontologies as input and outputs their alignment. As the authors state, an improvement would be to extract the word embeddings on a domain corpus especially for ontology alignment tasks.

An implementation of Word2vec can be found for ontology enrichment in Turkish language [25]. The vector representations are generated using the Turkish Wikipedia (Vikipedi). In detail, these embeddings are created based on the metadata of pages of Vikipedi and used as a golden standard to evaluate their approach of ontology enrichment.

Moreover, semi-automated ontology induction methods exist which are using word embeddings along with the contribution of domain experts [26]. This approach aims to exploit concepts, hierarchy, properties and relations from unstructured data like text. In order to generate the embeddings CBOW, Skip-Gram and Glove methods are applied on the text separately. Then, hierarchical clustering is performed on these embeddings. Finally, with the help of domain experts, who evaluate the clusters, the ontology components are created.

Wnet2vec is a framework where vector representations are generated from WordNet [27]. In order to obtain the embeddings a part of the Princeton WordNet is used. A semantic space is created from a semantic network. In detail, an ontological graph is converted into an embeddings matrix where it is used to measure the semantic similarity between words. Its performance is also compared with word2vec. Before creating the embedding matrix, principal component analysis is performed in order to reduce the size of the vectors to 850 dimensions. Word2vec embeddings are generated by training over a 100-token collection of texts. After comparing both methods on Simlex-999 dataset, wnet2vec outperforms word2vec. This happens due to the fact that wnet2vec embeddings are based on internal language resources. This means that WordNet captures the relations among words since it is crafted by experts while word2vec searches for this relation statistically on a given context window. Thus, word2vec may not be able to capture some of the lexical knowledge in the minds of speakers.

There are also frameworks that take advantage of pre-trained word embedding models. Specifically, the DeepAlignment framework refines pre-trained word embeddings of entities so that they can be used in ontology alignment tasks [28]. To do so, counter-fitting method is applied [29]. This method uses semantic lexicons in order to extract synonymy and antonymy relations. Then, these relations are used in order to refine the pre-trained word vector representations. This method enables the infusion of domain knowledge in

the ontology embeddings which are generated. In particular, the counter-fitting method generates new vector representations based on synonymy and antonymy constraints using a non-convex optimization method. The ontology alignment is accomplished using the Stable Marriage algorithm over the pairwise distance between the embeddings of the ontological entities

In their study D. Gromann and T. Declerck examine the use of pre-trained word embeddings of four different languages in an ontology matching process [30]. The pre-trained word representations are obtained from three repositories: Polyglot [31], FastText [32] and Word2vec. Regarding the ontology alignment, two ontologies describing Industry classification systems are used, the Classification Standard (GICS) and the Industry Classification Benchmark (ICB). This approach focuses on the labels of the ontology entities. Cosine similarity is used as a metric in order to measure the similarity among the vector representations combined in the previous step.

Furthermore, another method proposed tries to predict the RDF Schema from the instance level of a graph by using pre-trained word2vec embeddings [33]. The author of this work aims to visualize the RDF schema by combining characteristic sets with embeddings. This approach takes place on the instance level of a RDF graph rather than the schema of the graph. In fact, the ontology of the graph is considered loosely defined and thus it left out of this framework. On the other hand, this approach tries to generate the ontology behind the data. A characteristic set consists of the relations that a subject holds in a RDF graph. Moreover, this study uses two methods regarding the embeddings generation: a) Pre-trained word2vec embeddings and b) RDF2vec Embeddings generated by the instance level of the RDF graph.

RDF2vec is a tool for generating vector representations from a graph. Most of its applications are on the instance level of a graph [34]. The aim of the RDF2vec framework is the projection of the latent representation of entities into a lower dimensional feature space focusing on RDF graphs. Firstly, the RDF graph must be converted into a set of sequences of entities. To do so, two approaches are used: a) a modification of Weisfeiler-Lehman Subtree RDF graph kernels and b) graph walks.

In graph walks, the breadth-first algorithm takes place. Specifically, in the first iteration the paths are generated by exploring the direct outgoing edges. Then, until a finite number of iterations all the connected nodes explored according to the edges from the previous step. Finally, by combining the sequences of the nodes the set of sequences is created.

Regarding the modification of Weisfeiler-Lehman Subtree graph kernel this algorithm uses the Weisfeiler-Lehman test in so the number of subtrees shared between a number of graphs is calculated. In order to employ this approach to RDF graphs two modifications take place. Firstly, the edges of the graphs are directed. Secondly, the implementation of tracking the neighboring labels for the sake of consistency. Then, word2vec is employed in order to learn the embeddings. These entity representations can be used in various machine learning tasks like content-based recommender systems, link predictions, type prediction and graph completion.

RDF2Vec framework is also used in classification of ontology alignment changes [35]. It is implemented in order to identify the changes in ontologies which may interfere with ontology alignment. For this reason, RDF embeddings and classification algorithms are used. Identifying whether a change on an ontology may interfere on the alignment between this particular ontology with another one is a difficult task which relies on a detailed set of rules. This set of rules in most cases are not applicable to other domains so re-using them is not always valid. In this approach, the embeddings are generated from the ontologies as well as their alignment. In the next step, these vectors are used to train a classifier in order to identify whether a change affected the alignment or not. This approach is universal since it does not depend on a set of specified rules and could be applied in any alignment.

Implementations of using RDF2Vec in ontology alignment tasks can be found also in the private sector [36]. In detail, schema matching processes are exploited by creating a framework for fully automated schema matching regarding the private sector. Schema matching and ontology matching can be considered as analogous terms since all the techniques for ontology matching are applicable to schema matching. Finally, by implementing RDF2Vec the concept vectors can be obtained. It is worth mentioning that this study tries to incorporate the semantic techniques and approaches into the business world.

ALOD2Vec matcher aims to merge RDF2Vec with the Semantic Web technologies [37]. Specifically, it tries to align two ontologies by using WebIsALOD as an external source of knowledge [38]. WebIsALOD consists of the hypernym relations of LOD. These relations are extracted using the Common Crawl tool. These kinds of relations are crucial for the Semantic Web Ontologies. In particular, hypernym relations incorporate tail-entity relations for instance level data. Vector representations are generated on WebIsALOD dataset using RDF2Vec and used as background knowledge for the ontology alignment

task. Regarding the alignment process, the descriptions of the ontologies are obtained and matched using simple string-matching techniques. The ontology entities which are not matched in the first step are then matched to the background knowledge embeddings which are pre-computed as mentioned above. The label of the ontology entities is matched to the concepts of theme embeddings. After that, several candidates are obtained which are ranked according to a similarity measure.

Regarding the Semantic Web technologies, users in LOD can utilize the SPARQL mechanism in order to submit queries on a great collection of linked RDF datasets. They often come across with empty answer sets since their query may not be able to retrieve any particular information. For this reason, a framework is proposed which aims to solve this issue by generating graph embeddings on the instance level of a knowledge graph [39]. To do so, graph embeddings are generated from the RDF graph using TrasnE [40]. This model is chosen since it is capable of persevering the correlations among entities and relations. This way the inherent structure of the knowledge is preserved and projected to the vector space in contrast to neural-language based models like RDF2vec which cannot capture the relations between two entities while generating the embeddings. After generating the embeddings, this framework focuses on the SPARQL query which returns an empty set. In detail, the query is divided into parts. Using these parts along with the RDF embeddings it calculates alternative answers and provides them back to the user. Finally, each alternative answer is accompanied by an alternative query which may help users to refine their original queries.

The TransE framework generates embeddings by modeling the relations of a knowledge graph to translations in the embedding space. It mainly focuses on instance level data rather than ontology concepts. The main goal of TransE is to provide a way in order to complete missing relations in a graph without requiring any extra knowledge. In detail, it examines the hierarchical relationships since they have a crucial role in the majority of most knowledge graphs. As the authors state, they try to capture the key relations of a knowledge graph. In knowledge graphs, the relations that are 1-to-1 between different types of entities are of great value since they efficiently improve modeling. An example could be “author of” where an author and book are connected via such a relation. TransE focuses on such relations by including them in its process. Another key feature of this framework is its scalability. TransE is used in modeling the WordNet and Freebase knowledge bases in order to perform link prediction tasks.

TransE is also used in a framework which aims to produce high quality vector representations for the instance level of a network based on the relations among the concepts of an ontology [41]. This framework mainly focuses on Datalog± rather RDFS or OWL languages. Various types of vector space embeddings on the instance level of a knowledge graph are explored. The generated relation embeddings are obtained using TransE while DistMult is used to form the regions [42]. In addition, this framework models the relations of an ontology as constraints to these regions. It is considered that the relations of ontology formulate rules which can be translated to spatial constraints. After that, the framework checks whether an embedding captures the rules of the ontology. If this is the case, then this embedding is called a geometric model of the ontology. This prerequisite ensures that high quality representations are obtained from a knowledge graph.

Another framework worth mentioning which focuses on the instance level of a network is node2vec [43]. This framework generates embeddings for nodes in a network. It focuses on the instance level of networks rather than the ontology level. The goal of this approach is to create embeddings that maximize the likelihood of preserving network neighborhoods of nodes. In other words, it tries to capture the communities of the network. Creating such a framework is crucial since it could be used across several supervised machine learning tasks. Until recently, a common method to extract features from a network was with human intervention since it required domain specific knowledge. On the opposite side, node2vec extracts the communities of the network by implementing a variety of biased random walks. Node2vec generates the feature vectors that embed nodes in the same neighbor but also captures the information regarding nodes which have the same role in a neighbor (e.g. central node). Moreover, an extension of the Skip-gram method is applied where the goal is to optimize an objective function. To achieve that, it utilizes stochastic gradient descent with backpropagation on a neural network with one hidden layer. Another key feature of node2vec is that its major phases are parallelizable, making it a scalable graph embedding algorithm. After their generation, the embeddings are used in order to solve link prediction and multi-label classification problems.

The Global-RDF embeddings come in contrast to RDF2vec methodology [44]. This method introduces a new embedding approach inspired by Global Vectors (GloVe) on the instance level of a knowledge graph. In this approach, the vectors are generated based on the global patterns rather than the local patterns like paths, walks or kernels. To achieve

this, a co-occurrence matrix is built from a graph regarding the instance level. In order to compute this co-occurrence matrix an algorithm based on Personalized PageRank called Bookmark-Coloring Algorithm is applied. This framework allows a faster computation of the Personalized PageRank. The experimental results indicate an equivalent performance in comparison with RDF2Vec which is based on local patterns.

EmbedS framework takes into consideration both the ontology and the instance level of a graph in order to generate instance level embeddings [45]. In most approaches the ontology of a graph is often not taken into consideration. This results in cases where simple constraints referred to in an ontology are left aside (e.g. a book cannot be a friend of institution). Thus, it cannot take advantage of the rich and useful metadata information provided by an ontology. EmbedS is applied on a particular RDF dataset along with its ontology in the same vector space. This enables the calculation of ontological constraints and thus it could be used for further actions like feeding up a machine learning model.

Another approach which combines embeddings from both the instance and the ontology level of a knowledge graph is the JOIE framework [46]. It consists of two main components: a) the Cross-view Association Model and b) the Intra-view Model. The Cross-view Association Model is incorporated by two methods. The first method is called cross-view grouping and presumes the instance and the ontology level of a knowledge graph so they can be embedded into the same vector space. On the other hand, the second method is called cross-view transformation and uses two separate embedding spaces which will be combined by mapping the instance level embeddings to the ontology vector space. The Intra-view model is responsible for maintaining the structure in both the instance and the ontology level views. Two approaches are used: a) the Default Intra-view model and b) the Hierarchy-Aware Intra-view Model. The first one generates embeddings from the triplets while the second one focuses on the ontology. In particular, the Hierarchy-Aware Intra-view Model concentrates on the meta-relations provided by an ontology like “subclass of” since they provide crucial knowledge regarding the hierarchy structure. Finally, the framework combines the two main components mentioned above, in a joint function.

In comparison to the JOIE framework, HONOR combines these embeddings in the ontology vector space in order to perform a supervised normalization for entities extracted from text [47].

Another novel implementation of embeddings regarding the instance level is introduced in KGvec2go [48]. In detail, KGvec2go offers embeddings as a service. The vector

representations are obtained on the instance level entities via a Web query rather than generating the embeddings from a knowledge graph. Simply, it can be understood as pre-trained graph embeddings from four popular knowledge graphs. In particular, this framework generates the embedding using RDF2vec but with a more efficient walk generation process. As mentioned above four knowledge graphs are exploited: DBpedia, WebIsALOD, Wiktionary and WordNet. Combining these knowledge graphs, it could be extremely beneficial for various applications. The innovation of this framework comes to the way it provides embeddings calculations online. Users can use a Web API in order to consume these embeddings. This API allows embeddings to be used by less powerful devices such as tablets or smartphones. Specifically, similarity and N-closest entities calculations could be performed online by only providing the name of concepts rather than a URI. Users also have the ability to download the pretrained embeddings and merge them in their applications according to their needs. This framework is also available for running in an HTTP server.

There are numerous RDF embeddings approaches and knowledge graph embeddings methodologies but most of these efforts focus on creating embeddings on data instances rather than the ontology. OWL2Vec framework focuses directly on the ontologies. It relies on a modified version of RDF2Vec [49]. In detail, this framework consists of three stages: a) the Ontology projection, b) the Walk Strategy and c) the Concept Embeddings. In the Ontology projection the ontology is projected into a graph where the nodes represent concepts while the edges represent possible relations.

While in the Walk Strategy the walk on the ontology graph takes place. One of the key features of this approach is that the weights on the edges could be adjusted in order to give more significance to the taxonomic relationships or to the object properties.

Finally, the Concept Embeddings stage is using Word2vec and FastText in order to compute the concept embeddings. This method is flexible due to the fact that different concept embeddings could be created from different types of corpora sentences.

On2Vec aims to predict the relations among ontological entities in an ontology using embeddings generated from an ontology graph [50]. The main characteristic of ontology graphs is that they have comprehensive semantic relations. These relations hold facts regarding the hierarchy as well as the transitivity and symmetry of properties. The motivation behind the proposed model is a more pliable approach on generating ontology graph embeddings which do not rely on text corpora like other studies [51][52][53][54][55].

On2Vec is a translation-based graph embedding model that aims to identify the comprehensive ontology semantic relations. In detail, it consists of two main components: a) the Component-specific Model and b) the Hierarchy Model. The first one implements specific projections on the source and target concepts so the relational properties could be preserved. The second one applies a perceptive learning process on the hierarchical structure of the relations. In their experiments, the ontologies of Yago, ConceptNet and DBpedia OWL are used. Finally, after generating the ontology embeddings a relation prediction is performed between concepts.

Until now, there are few approaches which introduce ontology embedding to Neural Networks. In their study A. Benarab, F. Rafique and J. Sun introduce a methodology where the ontology embeddings are generated using multiple neural networks along with a neural network autoencoder [56]. The aim of this methodology is to create low representations of ontological entities in order to be ready for machine learning and deep learning procedures. In particular, this approach focuses on the semantic relations. This multi-neural network model aims to generate an object based on the subject and the relation of a triplet. For this reason, the relations are modeled in a neural network. Having multiple neural networks for a number of specified relations will form the model. One of the limitations of such an approach is the creation of sparse vectors. To deal with this issue an autoencoder neural network is used. The autoencoder is an unsupervised method consisting of an encoder and a decoder. It is a five-layer neural network with three hidden layers. The goal is to minimize a loss function. The full model consisting of the multiple neural networks along with the autoencoder is demonstrated in Figure 3: The architecture of the Multiple Neural Networks along with a Neural Network autoencoder as proposed in [56].

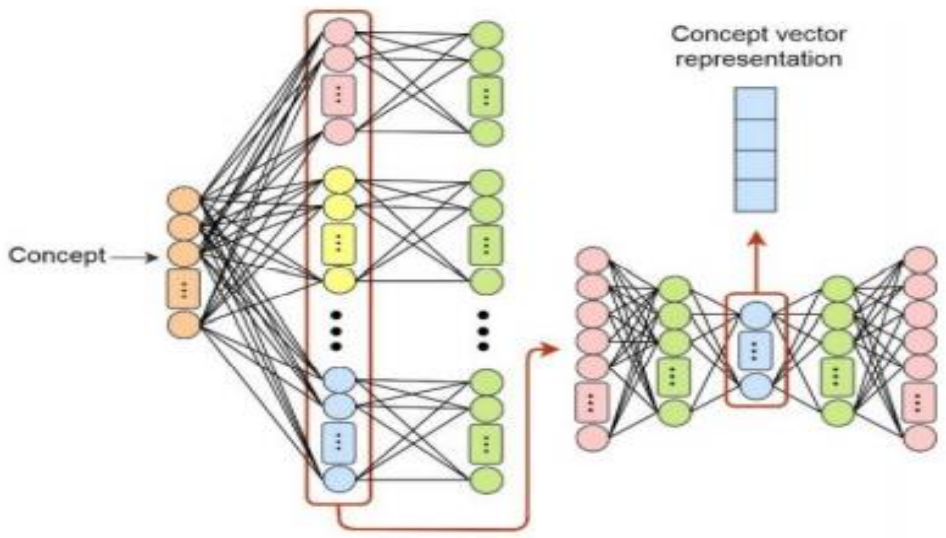


Figure 3: The architecture of the Multiple Neural Networks along with a Neural Network auto-encoder as proposed in [56].

Evaluating the quality of an ontology embedding is still in a premature stage. Specific evaluation metrics are proposed regarding the quality of ontology embeddings in this study [57]. It is stated that there is no systematic approach for evaluating the quality of embeddings in general. The focus of this work is to provide an "intrinsic metric" to solve the above issue. In detail, there are three aspects: a) the categorization, b) the hierarchical and c) the relational aspect. The first aspect focuses on both the instance level of the data and the ontology. The aspects 'b' and 'c' take only into consideration the ontology. Specifically, three metrics are proposed regarding the hierarchical aspect which focus on the ontology classes and subclasses rather than the relations: absolute semantic error, semantic Relatedness metric and Visualizations. In the relation aspect, two metrics are introduced: sectional preference and semantic transition distance. These metrics examine the relations among the ontology concepts.

2.3 European Union's Business Collections

In this part of this chapter the Business Collections[2] retrieved from the official site of European Union's Publications Office are analyzed. Business Collections are sets of vocabularies, ontologies and application profiles aiming to provide interoperable solutions to both the public and the private sector in the European Union. Apart from the Business Collections, the "Asset Description Metadata Schema Application Profile" (ADMS-AP) and the schemas in "Schema.org" are exploited.

Akoma Ntoso for European Union (AKN4EU)

AKN4EU provides a common structure for EU Legislative Documents. Having a common structure for legislative documents is crucial regarding interoperability. Apart from that, it creates a productive environment for interinstitutional legislative processes among institutions in the EU. AKN4EU is based on XML and aims to enable the exchange of legal documents. Specifically, it is constructed according to Akoma Ntoso and OASIS standards. Akoma Ntoso provides a set of representations in XML format of parliamentary, legislative and judiciary documents. OASIS goal is to create a path to standardization in international policy and procurement via open source and open standards. In its current version AKN4EU accommodates legal acts adopted through the ordinary legislative procedure and their legislative proposals. The future versions aim to develop interinstitutional standards for the exchange of structured content.

Book Interchange Tag Suite (BITS)

The BITS ontology has been selected for the production of general publications by the Publications Office of the European Union. This ontology is based on XML. Publishers can use this ontology in order to exchange book content. In detail, it defines the elements and attributes which are used to describe textual and graphical contents.

[2] https://op.europa.eu/en/web/eu-vocabularies/business-collections

Common Data Model (CDM)

The CDM ontology is constructed according to the Functional Requirements for Bibliographic Records (FRBR) model. Based on FRBR principles CDM uses OWL and RDFS technologies in order to represent the relations and the attributes among the ontology entities. Using the VOWL component in Protégé tool[3] the graph representation of the ontology is generated.

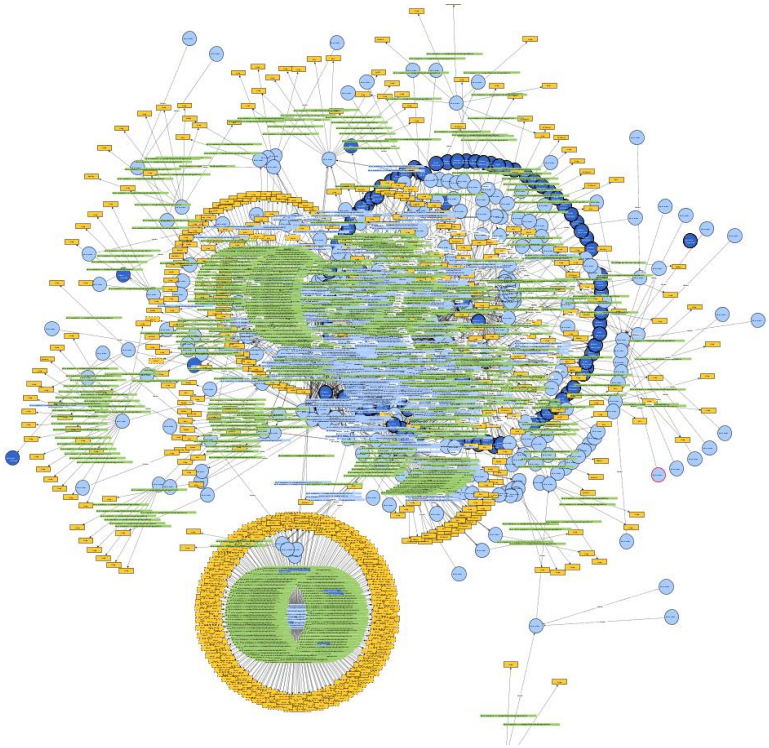


Figure 4: CDM Ontology graph generated using Protégé tool.

FRBR entity relationship model captures all the information regarding the material and tasks that are associated with bibliographic resources. In simpler words, it tries to model the bibliographic universe by offering a structurer among bibliographic concepts [58].

[3] https://protege.stanford.edu/

Core Vocabularies

The Core Vocabularies form a collection of re-usable data models. They capture the main attributes of an entity in a context-neutral approach. They are used in various scenarios. In particular, the Core Vocabularies constitute the starting point of information systems development. They are used for designing the conceptual and logical data models of a new system which also denotes their extensibility. Furthermore, they support a better information exchange among systems. In addition, they are used in various data integration processes between legacy systems which use different data models. Another great use of these vocabularies concerns the publication of Open data. They are used as a common export format among several portals. There are six Core Vocabularies. The Core Business Vocabulary models the characteristics of a legal entity. These characteristics may consist of the name, address activity etc. of a legal entity. The Core Location Vocabulary aligns with the INSPIRE data specifications and captures the main characteristics of a location. The Core Person Vocabulary models the characteristics of a person (e.g. name). The Core Public Organization Vocabulary designates the public organizations in the EU. The Core Public Service Vocabulary models the characteristics of services offered by public administrations e.g. title, description, inputs, outputs, providers, locations, etc. The Core Evidence and Criterion Vocabulary includes the means and the principles that must be held by a private entity in order to participate in public procurement.

The Digital Europa Thesaurus (DET)

The DET aims to model the web content in order to be retrieved, managed and aggregated across the European Commission's public communication. It reuses EuroVoc concepts along with concepts from other resources in order to describe web content. The DET can be characterized as multilingual thesaurus.

European Legislation Identifier (ELI)

The ELI framework provides as standardized format for online legislation metadata. Having a common format among these information enables the access, exchange and reuse across the Member States and organizations of the EU. In detail, the ELI ontology creates

a model regarding the online legislation metadata exchanging. The legislation description is based on FRBR principles.

eProcurement

The use of electronic means regarding the transactions and communications for buying supplies and services to be used by public sector organizations is called eProcurement.

This collection aims to the migration of public procurements to eProcurement by setting the standards for these procedures in the digital era. In detail, TED XML schemas along with authority tables and taxonomies are used to form this standard. Tenders Electronic Daily (TED) is the online version of the 'Supplement to the Official Journal' of the EU, dedicated to European public procurement. Publishing on TED requires the use of three schemas: a) the Reception, b) the Internal and c) Publication schema. The Reception schema includes all the forms that must be filled by all senders. The Internal schema is used inside the Publication Office. Finally, the Publication schema is used to publish the notice on TED's website.

Europass

In order to understand the skills and qualifications but also improve the transparency inside the EU, the Europass framework is created. This framework is a set of web-based tools and information that aims in better communication of skills and qualifications. It consists of a collection of Vocabularies. They are mainly used in Europass CV template, Europass Digital Credentials etc.

European Skills, Competences, Qualifications and Occupations (ESCO)

ESCO is the European multilingual classification of Skills, Competences, Qualifications and Occupations. It models the occupations, skills, qualifications and their relations that are relevant in the EU labor market. Furthermore, it offers job mobility since it creates a standard among the concepts mentioned above across boards.

The European Science Vocabulary (EuroSciVoc)

The EuroScivoc is a taxonomy created by the result of CORDIS. It is organized using semi-automatic Natural Language Processing techniques. This is a multilingual

taxonomy containing over 1000 categories, which are described using relevant keywords derived from CORDIS, in 6 languages. Specifically, it is created to be used as a reference vocabulary for the Open Science community.

Formalized Exchange of Electronic Publications (Formex)

The exchange of data among the Publication Office and its contractors is done by using the Formex format. Formex models the logical markup for documents published in the Official Journal of the European Union.

IMMC Core Metadata

This data model defines a set of the minimum metadata that are needed in the legal decision-making process. It enables a standard approach regarding information exchange between the institutions of the Publication Office.

Official Journal Electronic Exchange Protocol (OJEEP)

The OJEEP is a protocol responsible for the data exchange among the production system for the Official Journal of the European Union (PlanJO) and the printing contractors of the Official Journal. To accomplish its goal, this protocol consists of various scalable schemas which set the business requirements while enabling flawless information exchange.

OP Core metadata element set

This metadata model is created by the Publications Office of the EU and it is based on the Duplin Core metadata set. In particular, any resource published by the Publications Office should contain the 16 elements of OP core set.

DCAT Application profile for data portals in Europe (DCAT-AP)

The DCAT-AP is a specification for datasets in Europe. It is created according to Data Catalogue vocabulary (DCAT). DCAT is an RDF vocabulary which enables the use of a standard model so the metadata could be aggregated in multiple catalogs. In detail,

datasets using DCAT vocabulary are more discoverable since they could be retrieved by using the same search engine across multiple data portals. DCAT Application Profile as mentioned above is specification for metadata aiming to assist semantic interoperability. To achieve this goal, it reuses EuroVoc metadata vocabulary and mappings to existing vocabularies like Dublin Core. In other words, DCAT-AP provides a model which ensures consistency regarding the description of metadata and supports two main groups: a) the Data Reusers and b) Data Providers. The first group benefits from the fact that datasets can be easily retrieved because of the quality of their metadata. On the other hand, since the metadata is available across multiple data portals this enables availability at a low cost for the Data providers.

Asset Description Metadata Schema (ADMS)

The ADMS vocabulary is created in order to describe semantic interoperability solutions. There are three main stakeholders who use this vocabulary: a) the Solution Providers, b) the Content aggregators and c) the ICT developers. The first group is able to share and standardized metadata across platforms enabling discoverability. The Content aggregators can retrieve these metadata and provide them in a single point of access. This enables the ICT developers to search on one access point in order to explore interoperability solutions. The ADMS-AP extends the use of ADMS by providing solutions on political, legal, organizational and technical interoperability layers. A newer version of ADMS is also available. The main feature of the new version is the alignment with DCAT since it utilizes the DCAT classes rather than ADMS classes.

Schema.org

Schema.org is a shared vocabulary enabling developers and engineers to take advantage of existing schemas making their applications interoperable [4]. It is founded by Google, Microsoft, Yahoo and Yandex which are well known key players in the web. Developing a schema is done by an open community process mainly using GitHub[5] or by email. Vocabularies in Schema.org support various encodings such as RDFa, Microdata and JSON-

[4] https://schema.org/

[5] https://github.com/schemaorg/schemaorg

LD. One of the benefits of using such vocabularies is that they can be extended to cover developers' needs regarding their applications. Currently, Schema.org consists of 836 Types. Types are equivalent to Classes. Each Type holds two sets of properties: a) Properties from each Type and b) Properties from Thing. Regarding the first set of properties, they are created in order to describe a specific Type. Properties from Thing type, are common properties for all Types since every Type is a Thing.

3 Material and Methods

This chapter describes the methods implemented for text preprocessing, embeddings generation, feature extraction and finally the models used for the task of classification.

3.1 Introduction

The goal of this thesis is to examine the implementation of machine learning models with emphasis on word vector representations in the field of ontology engineering while focusing on the European Unions' ontologies. Specifically, it tries to tackle the problem of link prediction in an ontology set. In detail, this thesis focuses on the 'sub-property' relation among the entities in the ontology set. This problem is encountered as a classification task since it focuses on the prediction of whether two properties are connected or not. In addition, the Ontology Scout Tool (OS Tool) is introduced which is an ontology search tool based on pre-trained vector representation aiming to help ontology engineers identify concepts and ontologies of their interest. This chapter is divided into two main sections. The first section refers to the sub-property link prediction in an ontology set while the next section refers to the proposed ontology search tool. Regarding the first section, the procedure of obtaining the dataset is described. Moreover, a presentation of the Pretrained Embeddings Models which are used in order to generate the vector representations takes place. Then, the pipeline of Preprocessing and Feature extraction is thoroughly explained. Last but far from least, the Machine Learning Models along with their parameterization are demonstrated.

Apart from the problem of sub-property link prediction, an ontology search tool based on vector representations is also proposed in this thesis. The OS Tool is oriented towards ontology engineers and aims to provide assistance regarding tracking their ontologies of interests. In detail, it acts as an information retrieval tool based on pre-trained vector representations which provides ontology engineers and experts with fundamental information for the tasks of ontology development and ontology alignment.

3.2 Sub-Property Link Prediction in European Union's Ontology Set

3.2.1 Dataset

The dataset used in this thesis is created by exploiting the official European Unions ontologies. Since there is not an available dataset the following pipeline is followed in order to generate a balanced dataset for the link prediction task. The pipeline includes the following steps which will be later further analyzed:

a) Ontology parsing
b) Sub-property pairs extraction
c) False Subproperty pairs generation
d) Candidate pairs generation

3.2.1.1 Ontology Parsing

In this step, the ontologies are loaded and parsed using RDFLib Python's library[6]. This library is specifically created for working with RDF files. In addition, RDFLib also includes useful APIs for parsing data files of type RDF/XML, N3, NTriples, N-Quads, Turtle, TriX, RDFa and Microdata. The vast majority of the ontologies provided by the official European Union portal are in RDF/XML and Turtle format. In some cases where RDFLib could not parse various ontologies, the Protégé tool[7] is used to transform these particular ontologies into another format. Specifically, this procedure took place for BITS ontology where it transformed from XML into Turtle format. Moreover, due to incompatibilities of "ADMS version 2.0" ontology and RDFLib library, the "ADMS version 1.0" is used instead. Since the goal of this thesis is to use machine learning models in order to predict the sub-property relation between two properties only the following ontologies which hold concepts which have the type of property are parsed and loaded: CDM, ELI, European Commission Conceptual Framework, DCAT, European

[6] https://rdflib.readthedocs.io/en/stable/

[7] https://protege.stanford.edu/

Qualification Framework SKOS AP EU, DET SKOS AP EU, the Core Vocabularies, BITS and ADMS Version 1.0.

After parsing the ontologies, a network is created which contains the ontologies in the form of a graph. In detail, the created graph contains 971961 triples.

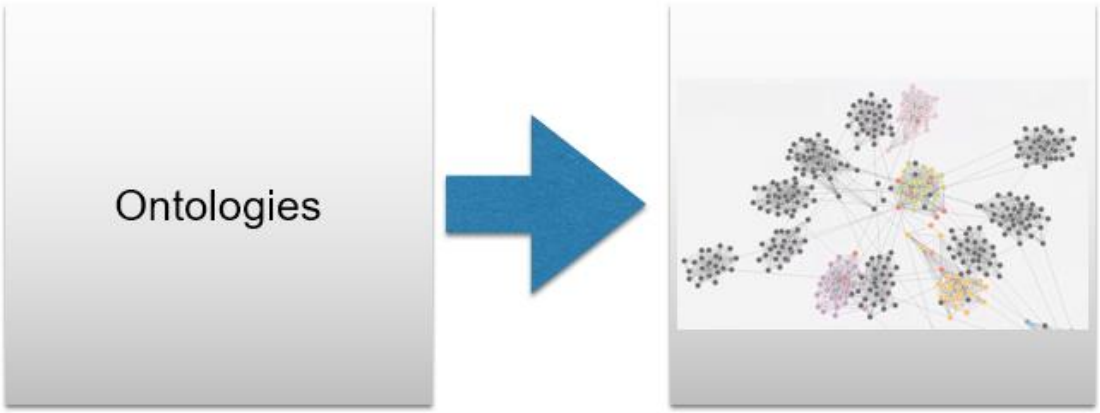


Figure 5: Parsing the ontologies creates a network of ontologies which forms a graph.

3.2.1.2 Sub-property pairs extraction

From the graph created like explained above, the sub-property pairs are extracted. To do so, the SPARQL Query Language[8] is used. Specifically, the query generated aims to retrieve URIs which are connected with the *"rdfs:subPropertyOf"* relation of the RDF Schema data modeling vocabulary[9].

All the URIS retrieved must hold at least one string value in order to generate word vector representations but also additional linguistic features which later will be used to train and test the machine learning models. For this reason, only Properties which have at least a label are retrieved. The results of the SPARQL query are loaded to a table. The total

[8] https://www.w3.org/TR/rdf-sparql-query/

[9] https://www.w3.org/TR/rdf-schema/

number of the retrieved property pairs is 1119. Moreover, an extra column is added named "Link" with value equal to 1 which indicates that the two URIs have a sub-property relation. Since it is a sub-property relation the URIs in the tale are labeled "super-property URI" and "property URI" accordingly.

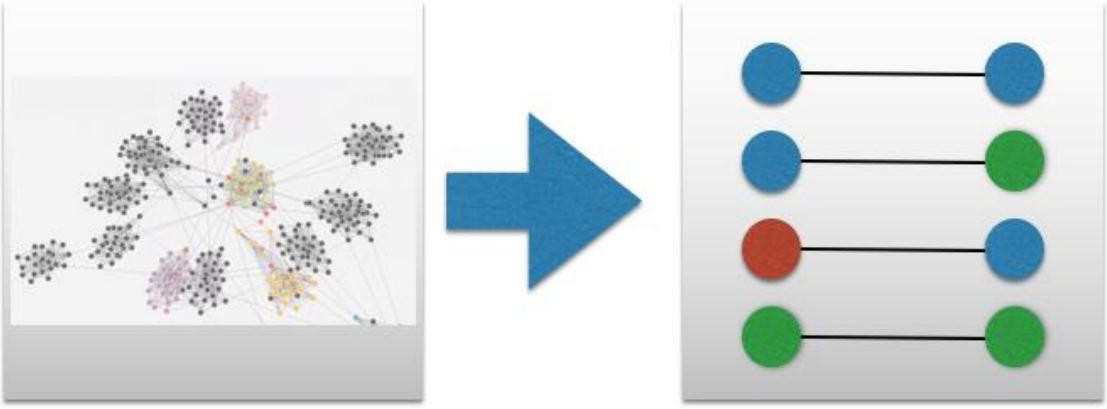


Figure 6: From the ontology graph the properties with the sub-property relation are extracted. These properties have *link* equal to 1 since it actually exists a connection between them.

	superproperty_uri	property_uri	link
0	http://publications.europa.eu/ontology/cdm#reviewed_by	http://publications.europa.eu/ontology/cdm#resource_legal_reviewed_by_case-law	1
1	http://publications.europa.eu/ontology/cdm#information_language_expression	http://publications.europa.eu/ontology/cdm#expression_publication_general_information_language_expression	1
2	http://data.europa.eu/eli/ontology#changed_by	http://data.europa.eu/eli/ontology#repealed_by	1
3	http://publications.europa.eu/ontology/cdm#objection_period	http://publications.europa.eu/ontology/cdm#event_internal_objection_period	1
4	http://publications.europa.eu/ontology/cdm#published	http://publications.europa.eu/ontology/cdm#question_parliamentary_with-debate_published	1
5	http://publications.europa.eu/ontology/cdm#implicitly_repealed_by	http://publications.europa.eu/ontology/cdm#resource_legal_implicitly_repealed_by_resource_legal	1
6	http://publications.europa.eu/ontology/cdm#modified_by	http://publications.europa.eu/ontology/cdm#legislation_secondary_modified_by_legislation_secondary	1
7	http://publications.europa.eu/ontology/cdm#version_notes	http://publications.europa.eu/ontology/cdm#work_dataset_version_notes	1
8	http://publications.europa.eu/ontology/cdm#number_procedure	http://publications.europa.eu/ontology/cdm#procedure_code_interinstitutional_number_procedure	1
9	http://publications.europa.eu/ontology/cdm#identifier	http://publications.europa.eu/ontology/cdm#eli	1
10	http://publications.europa.eu/ontology/cdm#implements	http://publications.europa.eu/ontology/cdm#measure_national_implementing_implements_resource_legal	1
11	http://publications.europa.eu/ontology/cdm#located_in	http://publications.europa.eu/ontology/cdm#period_located_in_unit_administrative	1
12	http://publications.europa.eu/ontology/cdm#main-activities_code	http://publications.europa.eu/ontology/cdm#notice_prior-information_main-activities_code	1
13	http://publications.europa.eu/ontology/cdm#consolidated_by	http://publications.europa.eu/ontology/cdm#resource_legal_consolidated_by_act_consolidated	1
14	http://publications.europa.eu/ontology/cdm#has	http://publications.europa.eu/ontology/cdm#event_case_has_type-procedure	1
15	http://data.europa.eu/54i#isAt	http://data.europa.eu/54i#isAccessibleThrough	1
16	http://publications.europa.eu/ontology/cdm#title_information_additional	http://publications.europa.eu/ontology/cdm#expression_title_information_additional	1
17	http://publications.europa.eu/ontology/cdm#completes	http://publications.europa.eu/ontology/cdm#resource_legal_completes_resource_legal	1
18	http://publications.europa.eu/ontology/cdm#comment	http://publications.europa.eu/ontology/cdm#dossier_comment	1
19	http://publications.europa.eu/ontology/cdm#year_procedure	http://publications.europa.eu/ontology/cdm#procedure_code_interinstitutional_year_procedure	1

Figure 7: After extracting the properties with the sub-property relation they are loaded to a table.

3.2.1.3 False Sub-property pairs generation

In this part, the methodology of creating the false pairs is analyzed. It is important to generate false pairs in order to train the machine learning models. Based on the actual pairs generated from the graph, the false pairs are generated. Firstly, a list of the unique super-property URIs is extracted. Then, a random choice among these URIs takes place where its URI has equal probability of being chosen. For the super-property URI selected a new random choice of a URI is performed excluding all the URIs where the super-property URI has an actual sub-property relation. Finally, an extra column named “Link” with value equal to 0 is added as indicating that the two URIs do not have a sub-property relation. The resulted table has 1119 lines in order to create a balanced dataset for training and testing the machine learning models in later sections.

	superproperty_uri	property_uri	link
0	http://publications.europa.eu/ontology/cdm#adopted-proposal	http://publications.europa.eu/ontology/cdm#resource_legal_uses_originally_language	0
1	http://publications.europa.eu/ontology/cdm#located_in	http://publications.europa.eu/ontology/cdm#summary_case-law_jure_id_celex	0
2	http://publications.europa.eu/ontology/cdm#official_journal	http://publications.europa.eu/ontology/cdm#measure_national_implementing_reference_commission	0
3	http://publications.europa.eu/ontology/cdm#pending-proposal	http://publications.europa.eu/ontology/cdm#event_legal_responsibility_of_institution	0
4	http://publications.europa.eu/ontology/cdm#adopts	http://publications.europa.eu/ontology/cdm#case-law_national_following_case-law	0
5	http://publications.europa.eu/ontology/cdm#date_processing	http://publications.europa.eu/ontology/cdm#signature_digital_signs_manifestation	0
6	http://publications.europa.eu/ontology/cdm#dossier_contains_event	http://publications.europa.eu/ontology/cdm#resource_legal_basis_for_resource_legal	0
7	http://publications.europa.eu/ontology/cdm#used_by_table_of_contents	http://publications.europa.eu/ontology/cdm#resource_legal_number_sequence	0
8	http://publications.europa.eu/ontology/cdm#reproduced_as	http://publications.europa.eu/ontology/cdm#resource_legal_id_local	0
9	http://publications.europa.eu/ontology/cdm#date_end	http://publications.europa.eu/ontology/cdm#work_dataset_date_modified	0
10	http://publications.europa.eu/ontology/cdm#mandatorily-addresses	http://publications.europa.eu/ontology/cdm#work_date_document	0
11	http://publications.europa.eu/ontology/cdm#volume	http://publications.europa.eu/ontology/cdm#question_parliamentary_asked_by_group_parliamentary	0
12	http://publications.europa.eu/ontology/cdm#abstract	http://publications.europa.eu/ontology/cdm#expression_subtitle	0
13	http://publications.europa.eu/ontology/cdm#cited_by	http://publications.europa.eu/ontology/cdm#resource_legal_application_deferred_by_case-law	0
14	http://publications.europa.eu/ontology/cdm#parent_document_ref	http://publications.europa.eu/ontology/cdm#expression_accompanying_material_uses_language	0
15	http://publications.europa.eu/ontology/cdm#affaire_year	http://publications.europa.eu/ontology/cdm#event_case_framework_procedural_identifier	0
16	http://publications.europa.eu/ontology/cdm#objection_period_extension	http://publications.europa.eu/ontology/cdm#signature_digital_date_signature_xades-a	0
17	http://publications.europa.eu/ontology/cdm#has_part	http://publications.europa.eu/ontology/cdm#manifestation_related_to_manifestation	0
18	http://publications.europa.eu/ontology/cdm#article_journal_related	http://publications.europa.eu/ontology/cdm#resource_legal_implemented_by_resource_legal	0
19	http://publications.europa.eu/ontology/cdm#requests_interpretation	http://publications.europa.eu/ontology/cdm#manifestation_official-journal_part_page_last	0

Figure 8: The table of false sub-property pairs. These pairs are generated from the table in created in Figure 7: After extracting the properties with the sub-property relation they are loaded to a table.

By concatenating both the sets which hold the actual and false URI pairs the dataset is created having 2238 lines.

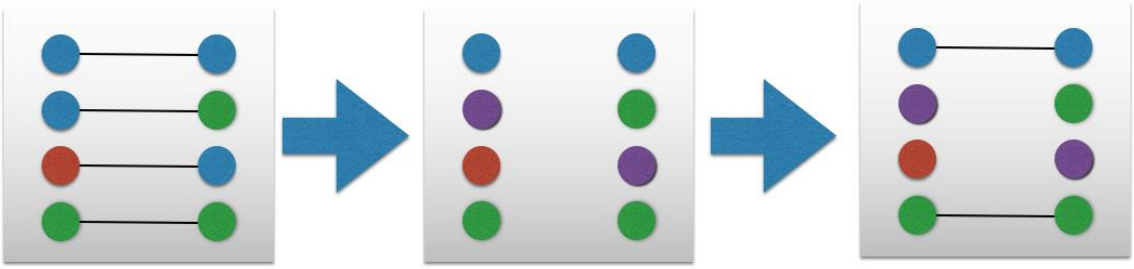


Figure 9: By concatenating the actual with false sub-property sets the dataset is generated.

3.2.1.4 Candidate pairs generation

In order to evaluate the performance of the machine learning algorithms on predicting the sub property relation between two properties we must have a test set. Creating the test set is done by splitting the dataset crafted above, into two sets: a) the training set and the test set. In detail, a stratified split takes place so both the training set and the test set are balanced regarding the "Link" class. The size of the test set is 20% of the original dataset including 448 pairs of URIs. Apart from the test set an extra set is generated for testing purposes. The candidate pairs are generated based on the URIs which are included in the test set. Concerning the candidate pairs generation, they are generated in respect to the word vector representations of the labels and comments of the URIs in the test set. For each unique "super-property" URI in the test set, 5 candidate property URIs are generated based on the cosine similarity of their label and comment vector representations. For this reason, Sentence-BERT is used which is a modification of the pretrained BERT network. More analytically, Sentence-BERT (SBERT) model utilizes siamese and triplet network structures to generate semantically meaningful sentence embeddings [59]. Generating the sentence embeddings for the labels and comments is done using the "distilbert-base-nli-stsb-mean-tokens" model which is optimized for semantic textual similarity and also has a balance in terms of speed and performance. For each unique "super-property" URI in the test set the label embedding is calculated while for all property URIs the embedding of their labels and comments are calculated. Each embedding generated using the SBERT model has 768 dimensions. Having such a number of dimensions comes with a high memory and computation cost. On the other hand, having a large number in vector representations provides a great detail for each sentence in the vector space which is crucial for the purpose of generating candidate pairs. For each unique "Super-property" the

cosine similarity of its label embedding along with the property label and comment embeddings are calculated. Then, the highest cosine between the “super-property” label and the property label and “super-property” label and the property comment is kept serving as a ranking measure. The top 5 property URIs along with the “super-property” URI form the candidate pairs and are appended in a table. The size of this table is 765 lines which is the number of the unique “super-property” URIs in the test set multiplied by five since this is the number of candidate pairs generated. Finally, the existence of relation between the candidate pairs is derived from the original test set and is demonstrated to a column named “Link” representing the relation of the two URIs accordingly. Figure 10 demonstrates the closest labels to “is motivated by” label in a two-dimensional vector space produced by performing principal component analysis to the vector representations transforming them from 768 to 2 dimensions.

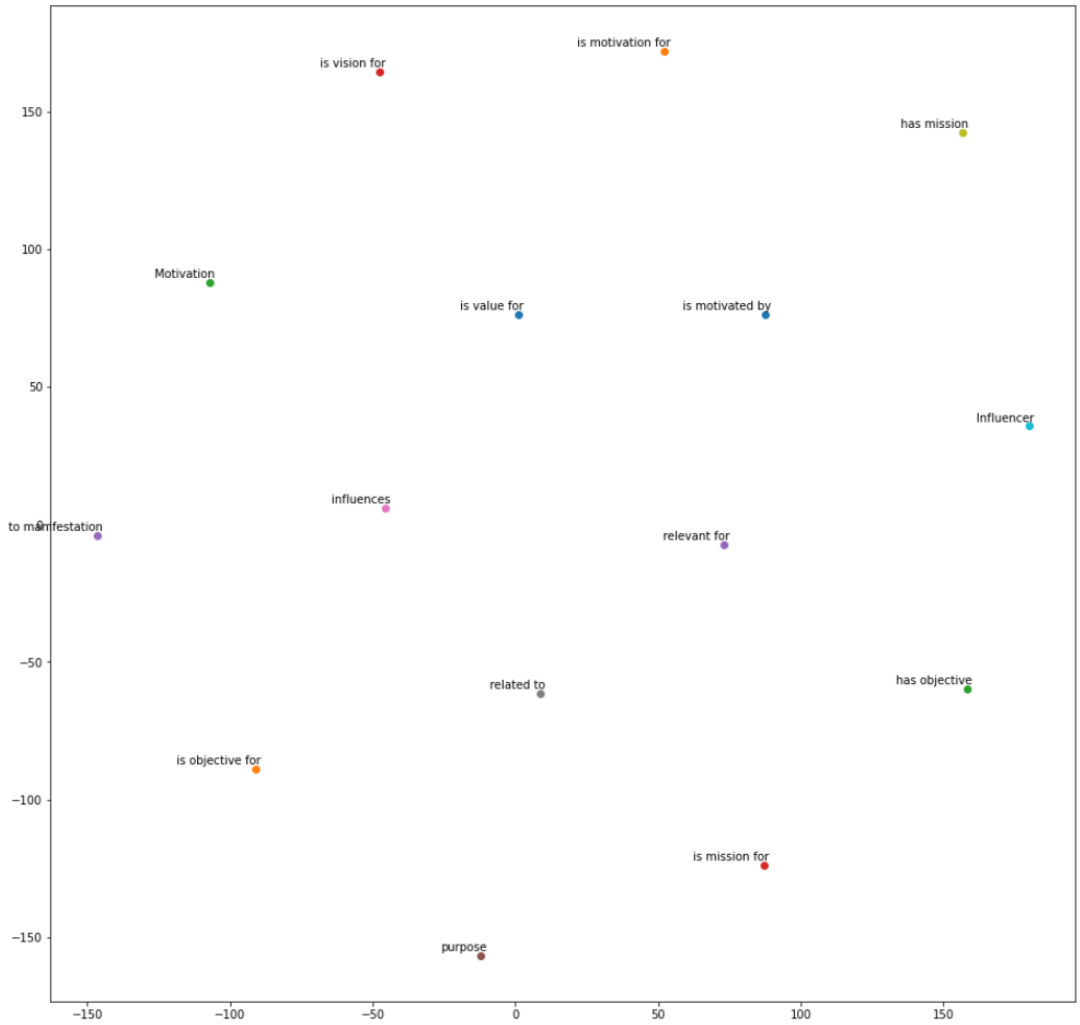


Figure 10: The closest labels to “is motivated by” in a two-dimensional space.

3.2.2 Pretrained Embedding Models

The purpose of this thesis is to examine the implementation of machine learning methods with emphasis on word vector representations in the field of ontologies. For this reason, 4 Pretrained Embedding Models are used. The implementation of these 4 different models results in 4 separate solutions for the problem of this thesis. In detail, for each pre-trained embedding model a different set of features is created. This approach enables the comparison among these models in terms of their accuracy and other measures in the link prediction task. While the embedding models may differ, the whole pipeline explained in the feature extraction section is common for all approaches.

As mentioned above this thesis takes advantage of 4 different pretrained embeddings models which are used to generate the embeddings for the labels and the comments of the properties in the dataset.

The following models are used in order to generate the vector representations:

a) Sentence-Bert b) spaCy c) Word2vec-Gensim d) Glove.

To extract the labels and comments for each property a SPARQL query is constructed which retrieves the necessary data. The result of this query is then passed in a table for further analysis as shown in Figure 11. Since vector representations as well as other linguistic features which are constructed in this thesis are based on Natural Language Processing techniques, the Properties retrieved must have a label while the existence of a comment is optional. For Properties which do not have a comment the value in “nan”. For any “nan” comment, its embedding will be a vector of zeros across all dimensions so it could be later used to calculate the cosine similarity between properties but also in machine learning tasks.

	uri	label	comment
0	http://data.europa.eu/54#Channel	Channel	nan
1	http://publications.europa.eu/ontology/cdm#resource_legal_year	Year of the legal resource	Year of the legal resource. DT_ANN. Relevant for legal analysis: true
2	http://data.europa.eu/eli/ontology#Version	Version	Formally defined as the set of skos:Concept in concept scheme eli:VersionTable
3	http://publications.europa.eu/ontology/cdm#publication_general_title_part	Title of part	Name of part or section of a work.\nMarc fields 700 p, 710 p and 711 n | How do fields 700 p, 710 p and 711 n differ?
4	http://publications.europa.eu/ontology/cdm#official-journal	Official Journal	The Official Journal itself
5	http://publications.europa.eu/ontology/cdm#summary_legislation_eu_id_doc_related	Natural number of related documents	
6	http://publications.europa.eu/ontology/cdm#case_court_affected_by_case_court	Case affected by case	This case affected by another case
7	http://publications.europa.eu/ontology/cdm#located_in	located_in	nan
8	http://publications.europa.eu/ontology/cdm#has_conclusions	has_conclusions	nan
9	http://data.europa.eu/eli/ontology#specificPropertyOf	specific property of	Indicates that a property is, in a generic sense, a more specific property of another one. This does not entail the same logical reasoning as rdfs:subPropertyOf.\nThis is used to indicate the alignement to Dublin Core properties, without redeclaring the Dublin Core property in Protégé, and without commiting to choose that a DubinCore property is either a datatype or an object property.
10	http://publications.europa.eu/ontology/cdm#concept_type_comment	Concept representing the type of a comment	Corresponds to the contents of the Procat field CI, coded with FD_Table 301
11	http://publications.europa.eu/ontology/cdm#submits_preliminary_question	submits_preliminary_question	nan
12	http://publications.europa.eu/ontology/cdm#procurement_public_heading	Heading of the procurement notice	
13	http://publications.europa.eu/ontology/cdm#date_entry-into-force	date_entry-into-force	nan
14	http://data.europa.eu/54#InformationCarrier	Information Carrier	A carrier of information. It can be non-electric like paper document, business card, or poster. \nIt can be electronic such as file, or combined such a business card or a digital badge.
15	http://publications.europa.eu/ontology/cdm#event_case_part1	Part 1 of the legal basis of the procedure	Part 1 of the legal basis of the procedure.\ntypical values PROT
16	http://publications.europa.eu/ontology/cdm#replaced_by	replaced_by	nan
17	http://publications.europa.eu/ontology/cdm#case-law_requests_opinion_resource_legal	Case law requests an opinion	2 cases (2012-07-31). Example: 62008CU0001: "request for opinion 12006E133 P1"
18	http://publications.europa.eu/ontology/cdm#resource_legal_validation_requested_by_case-law	Validation of legal resource requested by case law	Relevant for legal analysis: true
19	http://publications.europa.eu/ontology/cdm#has_media_type	has_media_type	nan

Figure 11: The labels and comments for all URIs are extracted in a table.

3.2.2.1 Sentence Bert

As mentioned above, the Sentence-Bert (SBERT) model is used to generate the candidate pairs which will later be used to test the machine learning algorithms. The SBERT model is also used to derive vector representations regarding the labels and comments of the properties which will be used for training the machine learning algorithms. Since the SBERT model is able to produce embeddings directly from a sentence, the labels and comments for each property are passed through that model generating their corresponding embeddings. The size of each embedding is 768 dimensions. For the properties which hold no comments a vector of zeros is produced for each dimension so they could be used in machine learning. In SBERT, the "distilbert-base-nli-stsb-mean-tokens" model is used which is optimized for semantic textual similarity. Specifically, this model is trained on the SNLI[10] and the Multi-Genre NLI[11] datasets.

[10] https://www.aclweb.org/anthology/D15-1075/

[11] https://www.aclweb.org/anthology/N18-1101/

3.2.2.2 spaCy

SpaCy[12] is a library focusing on the implementation of Natural Language Processing applications and pipelines. This library provides a variety of models in numerous languages like English, German, Greek, Japanese and others. Apart from the above, it also provides various Multi-language models which can be used for comparison among different languages. In this thesis the "en_core_web_sm" pretrained model is utilized which provides sentence embeddings directly. It is a multi-task Convolutional Neural Network trained on OntoNotes [60].

The number of dimensions of the embeddings generated is 96. Furthermore, spaCy sentence embeddings models handle internally text preprocessing operations like punctuation removal and lower-case transformation.

3.2.2.3 Word2vec Gensim

Gensim[13] is an open-source library that contains efficient implementations of Natural Language Processing functionalities for the task of topic modeling. With Gensim, the embeddings are generated using the "GoogleNews-vectros-negative300" pre-trained embeddings model. These embeddings are trained on Google News dataset and provide word vector representations of 300 dimensions. This model is capable of providing word and phrase vector representations than sentence embeddings [13]. Moreover, it does not include a text preprocessing pipeline. For this reason, all labels and comments are passed through a text preprocess function where lower-case transformation, punctuation and stop words removal take place. After that, they are ready to be used to generate the word vector representations. For the purpose of this thesis, in order to derive the sentence embeddings for each label and comment, the mean vector representation of the words is calculated. For example, if a comment is constituted by 10 words, the embedding for this comment will be the mean vector of these 10 words. This procedure results in an embedding of 300 dimensions which represents the whole comment.

---

[12] https://spacy.io/

[13] https://radimrehurek.com/gensim/models/word2vec.html

3.2.2.4 GloVe

Global Vectors is an unsupervised technique for generating word vector representations. It is based on global text statistics by deriving semantic relations among words from the co-occurrence matrix [14]. In this thesis the “glove.6b.300d” model is used which is trained on Wikipedia 2014[14] and Gigaword 5[15] datasets. The size of these vector representations is 300 dimensions. Since these embeddings come in the form of a dictionary, there is no text preprocessing pipeline and only provide word vector representations. For this reason, as mentioned above, all labels and comments are passed through a text preprocess function and sentence embeddings are generated by calculating the mean vector for each individual word vector representation.

After generating the vector representations regarding the labels and comments for each property, the junction of these vectors with the original data takes place.

superproperty_uri	property_uri	link	superproperty_label	superproperty_comment	property_label	property_comment	vector_superproperty_label	vector_property_label	vector_superproperty_comment	vector_property_comment
http://publications.europa.eu/ontology/cdm#pla...	http://publications.europa.eu/ontology/cdm#pro...	0	planned_adoption_period_date_start	NaN	Internal procedure needs impact assessment?	Do the better_regulation_requirements require ...	[-0.005068203, 0.14570312, -0.002947998, 0.07...	[-0.097717285, 0.155896, 0.01986084, -0.011450...	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...	[-0.09231949, 0.02649943, 0.010096232, 0.05042...
ttp://publications.europa.eu/ontology/cdm#adopts	http://publications.europa.eu/ontology/cdm#res...	1	adopts	NaN	Legal resource adopts legal resource	Relevant for legal analysis: true	[0.13769531, 0.25390625, 0.092285156, 0.0625, ...	[-0.07116699, 0.021484375, 0.023144532, -0.014...	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...	[-0.075891115, -0.07119141, -0.03845215, 0.014...
http://publications.europa.eu/ontology/cdm#is_...	http://publications.europa.eu/ontology/cdm#pub...	1	is_about	NaN	General publication is about concept op theme	NaN	[0.104599, -0.07714844, 0.17773438, -0.0570678...	[0.09679194, 0.06452288, -0.020891482, 0.04394...	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
ttp://publications.europa.eu/ontology/cdm#man...	http://publications.europa.eu/ontology/cdm#man...	1	Last page	Last page number of the OJ part related to a l...	Last page	DEPRECATED, Last page number of the OJ part r...	[0.15966797, 0.05078125, 0.02029419, -0.129028...	[0.15966797, 0.05078125, 0.02029419, -0.129028...	[0.009640164, 0.028211806, 0.038845487, -0.004...	[0.0526886, 0.017299106, -0.0109470915, 0.0204...
http://publications.europa.eu/ontology/cdm#ide...	http://publications.europa.eu/ontology/cdm#exp...	1	identifier_case	NaN	Case identifier	Case identifier in the title of a case law doc...	[0.072143555, -0.08105469, 0.061279297, 0.1583...	[0.072143555, -0.08105469, 0.061279297, 0.1583...	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...	[0.040563963, -0.0138549805, 0.055249024, 0.06...
http://publications.europa.eu/ontology/cdm#has...	http://publications.europa.eu/ontology/cdm#wor...	1	has_type	NaN	Type of dataset	This property indicates the type of dataset	[-0.04284668, 0.031433105, -0.00642395, 0.1236...	[-0.000056641, 0.041107178, 0.16915894, 0.1662...	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...	[0.059977215, 0.017120361, 0.033564845, 0.0879...
http://publications.europa.eu/ontology/cdm#add...	http://publications.europa.eu/ontology/cdm#ins...	1	addressed_by	NaN	Institution addressed by legal resource	NaN	[-0.1558293, 0.034057617, 0.12426758, 0.005859...	[-0.13040772, -0.0022888184, 0.07128906, 0.061...	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
http://publications.europa.eu/ontology/cdm#han...	http://publications.europa.eu/ontology/cdm#eve...	1	handed_over	NaN	legal event handed over by person	This legal event is handed over by a given per...	[0.1352539, 0.15234375, 0.07672119, -0.1229248...	[0.02878248, 0.013468425, 0.018269857, 0.0146...	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...	[0.057254676, -0.009329131, 0.016296387, 0.042...
http://publications.europa.eu/ontology/cdm#dec...	http://publications.europa.eu/ontology/cdm#cas...	1	declares_incidentally_valid	NaN	Case law declares incidentally valid legal res...	111 cases (2010-07-27). How do codes G and W d...	[0.019429525, -0.16633098, 0.0777181, 0.064941...	[-0.0355399, -0.0728193, 0.043422155, 0.048648...	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...	[-0.009094238, 0.0063552856, 0.08934021, 0.168...
ttp://publications.europa.eu/ontology/cdm#com...	http://publications.europa.eu/ontology/cdm#pro...	1	comment_converter	NaN	Comment converter	NaN	[-0.17614746, -0.14477539, -0.14404297, -0.329...	[-0.17810059, -0.1315918, -0.0670166, -0.29638...	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...

Figure 12: The final dataset including the actual and false sub-property pairs along with the embedding for their labels and comments.

[14] https://wiki.dbpedia.org/Downloads2014

[15] https://catalog.ldc.upenn.edu/LDC2011T07

3.2.3 Text Preprocessing

Text is a valuable source of information which can be found in all fields of science and technology. It is an unstructured form of data which may contain noisy content. Cleaning the text requires a solid understanding of the field, so the mining of the text could result in meaningful insights. For this reason, the implementation of text cleaning actions is crucial and must take place before feature engineering and machine learning modeling. Labels and comments of properties are in text form. Text processing is implemented by using Natural Language Toolkit (NLTK)[16] and Regular Expressions (RE)[17] Python libraries. The techniques utilized for this problem are presented as follows:

Lowercase transformation

Since Python is a case sensitive programming language all alphabetical characters are converted to lowercase. This conversion enables the models to capture that two words have the same meaning despite their differentiation in letter cases.

URLs removal

Numerous comments include external links which refer to specific sources. Nevertheless, for the purpose of this thesis they are noisy text rather than meaningful information. Hence, all external links starting with “https” or “http” are removed.

Punctuations removal

In the field of ontologies and especially the labels and comments of the properties in the dataset, punctuations do not have any significant importance. Thus, they are removed from all text fields.

Tags removal

There are cases where the comments of some properties may contain tags. An example of a tag is most commonly a word starting with “@”. Since they do not provide any

[16] https://www.nltk.org/

[17] https://docs.python.org/3/library/re.html

additional information regarding the actual Property but may refer to external sources they are completely removed.

Tokenization

Converting text into a list of words (tokens) is an essential part of text preprocessing pipelines. Having tokens is mandatory for feature engineering which will be used in machine learning models.

Stop-words removal

Stop-words is a set of words that are frequently used and do not provide any actual insights. In this thesis, the set of stop-words provided from the NLTK library is utilized in order to remove them from all labels and comments.

Stemming

Stemming is the process which recognizes and maintains the root of a word. This is extremely significant since words which share the same root after stemming are considered as the same. This process is applied in all labels and comments in advance of feature engineering. It is not applied for the generation of vector representations.

3.2.4 Feature Extraction

Length of labels

This is a basic feature which calculates the number of words regarding the labels of the properties.

Number of common words

This feature calculates the number of common words between two string values. In this thesis, the number of common words for the following pairs is extracted: “super-

property" label & property label and "super-property" label & property comment. The feature is an integer number which demonstrates the number of common words.

Ratio number of common words and length of labels

This ration combines the "Number of common words" and "Length of labels" features. In detail, it divides the number of common words between the labels by the average string length of both labels.

The features below are extracted for the following pairs: "super-property" label & property label, "super-property" comment & property comment, "super-property" label & property comment and "super-property" comment & property label.

Fuzzy Logic

Fuzzy logic is a form of multi-valued logic that deals with reasoning that is approximate rather than fixed and exact. Fuzzy String Matching is the process of finding strings that approximately match a pattern. For the purpose of this thesis FuzzyWuzzy[18] library's method "fuzz ratio" is implemented in order to calculate the edit distance between some ordering of the tokens in both input strings.

Jaccard Similarity

The transformation of a sentence into a set of words enables the use of Jaccard Similarity measure. It is the ratio of the size of the intersection divided by the size of the union for two sets.

[18] https://github.com/seatgeek/fuzzywuzzy

TF-IDF Cosine Similarity

Cosine Similarity is a measure of similarity between two non-zero vectors of an inner product space. Firstly, a TF-IDF transformation is applied in order to get real-valued vectors. Using the TfidfVectorizer[19], TF-IDF is implemented. Then the cosine similarity is measured.

Embeddings Cosine Similarity

The cosine similarity regarding the embeddings generated from the pretrained embedding model is calculated.

Embeddings Values

Since the number of dimensions of a vector representation is pre-defined, it enables the transformations of embeddings to separate features. Each value of an embedding is appended to a separate column creating a new feature. This approach allows the machine learning algorithms to take full advantage of the vector representations in the dataset.

3.2.5 Machine Learning Models

In this section, the machine learning models utilized are analyzed. In total nine models are trained and tested regarding the problem of this thesis. Specifically, the models are trained on the training set which is obtained as described in the Dataset section. A summary of each machine learning model follows:

K-nearest Neighbors (k-NN):

K-NN algorithm is an instance-based learner and relies on a distance metric for the classification. It belongs to the family of non-parametric models [61]. The only parameter that users should define is the number of nearest neighbors which are evaluated in order

[19] https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html

to assign a class a given observation. In this thesis, the number of nearest neighbors is set to 3.

Naive Bayes (NB):

NB constitutes a probabilistic classifier inspired by the Bayes theorem under a simple assumption which is that the attributes are conditionally independent. This method is widely used because of its simplicity and its generally good results. Every pair of features being classified is independent of each other. In this thesis Guassian Naive Bayes algorithm is implemented which assumes that the likelihood of the features is Gaussian.

Decision Trees (DT):

Like K-NN, DT is a non-parametric algorithm utilized for classification. In classification, the leaves of the tree demonstrate the class labels [62]. The branches demonstrate the path that an observation follows regarding the features which leads to a leaf. The minimum number of samples required to separate a node is set to two while there is no limitation for the maximum depth. This means that the nodes are scaled until all leaves are pure or include less than two samples.

Random Forests (RM):

RM is a special case of Bootstrap aggregating methods and belongs to the ensemble learning algorithms family. It is implemented by fitting a number of Decision Tree classifiers on the dataset. Regarding classification problems, the prediction of a class for an observation is the mode of the classes predicted by the DT [63]. The number of trees utilized for RM classifier is fifteen while their maximum depth is set to six.

Support Vector Machines (SVC):

SVC is a non-probabilistic supervised machine learning method. It is implemented by separating the classes in space by leaving the widest possible gap between them. During the testing phase the unseen observations are mapped to this space and assigned to a class according to the part of the gap they fall [64]. The regularization parameter must be set for this algorithm. In this model it is set by using a standardization method which subtracts the mean and scales to standard deviation.

Support Vector Machines Bagging:

Bootstrap aggregating or Bagging is an ensemble learning method aiming to improve the metrics regarding a machine learning algorithm by combining the predictions from multiple models [65]. This method is utilized by using ten Support Vector Machines models whoose configuration is explained above.

Neural Networks:

Multilayer Perceptron (MLP) belongs to the family of Neural Networks. In this thesis the MLP implemented uses a nonlinear activation function in order to classify data that are not linearly separable. Every node in a layer connects to all nodes in the following layer making the network fully connected. Neural Networks in general are structured in three main layers, the input, the hidden and the output layer. The number of hidden layers is set to 20 while the number of epochs is set to ten. Regarding the learning phase, the method of indirect optimization is used. Finally, 'Adam' optimizer and Rectified Linear Unit activation function are utilized.

Adaptive Boosting (Adaboost):

Adaboost is a boosting algorithm that concentrates on the observations that are harder to classify by an iterative process. Adaboost iteratively fits a classifier to observations which are misclassified by focusing on the weights specifically for these instances [66].

This method is implemented by using 100 models of the Adaboost-SAMME algorithm [67].

Extreme Gradient Boosting (XGBoost):

XGBoost is a relatively new algorithm in machine learning. It is a scalable machine learning system for tree boosting [68]. One of its key advantages is the computation speed and scalability over a single machine which is comparable to other methods using a distributed

set up. XGBoost utilizes a tree learning algorithm which handles sparse data. Then it applies a sketch procedure enabling instance weights in tree learning to be handled more efficiently.

3.3 Ontology Scout Tool

This thesis proposes Ontology Scout Tool (OS Tool) an ontology search tool which is based on pre-trained vector representations. OS Tool addresses to ontology engineers and experts aiming to develop an ontology or perform ontology alignment tasks. It is implemented by parsing a set of ontologies resulting in an ontology set. This procedure is similar to the Ontology parsing section mentioned above regarding the problem of sub-property link prediction. Since it is based on pre-trained embeddings, Sentence-Bert (SBERT) model is utilized. SBERT is used to derive vector representations regarding the labels and comments of all instances in the ontology network. These can later be used in the retrieving process. As explained in previous sections, SBERT is capable of producing embeddings directly from a sentence. OS Tool takes advantage of this feature since its users may use multiple keywords to a query. Then this query is transformed to an embedding using SBERT and it is compared with the embeddings generated from the ontology network in the vector space. Finally, a sorted list of the instances is retrieved according to their cosine similarity with the query. The size of each embedding is 768 dimensions. Having such a number of dimensions provides detailed vector representations but increases the computational cost of this tool. For instances which do not hold any label or comment, a vector of zeros is produced for each dimension so they could be used in machine learning. As in the previous section, the “distilbert-base-nli-stsb-mean-tokens” model is utilized since it is optimized for semantic textual similarity. A prototype of OS Tool is demonstrated in the Experimental Results Section.

4 Experimental Results

This section describes the experiments conducted for the proposed method of sub-property link prediction in the EU's ontology set. Moreover, the Ontology Scout Tool (OS Tool) prototype is introduced. Regarding the link prediction problem, the different embedding techniques described in the Pretrained Embedding Models section are compared. As explained in Material and methods, 4 different approaches are implemented based on the embeddings models used. For each approach 9 machine learning models are trained and tested regarding the problem of this thesis. Furthermore, the results obtained are concerning the test and candidates set. Since it is a classification problem all methods are compared in terms of accuracy and precision for the above two sets. Accuracy and precision are calculated as shown below:

$$Accuracy = \frac{Number\ of\ correct\ predictions}{Total\ number\ of\ predictions}$$

$$Precision = \frac{Number\ of\ pairs\ predicted\ having\ the\ sub\ property\ relation}{Actual\ number\ of\ pairs\ having\ the\ sub\ property\ relation}$$

Regarding the candidates set, the “Mean Average Precision” is measured for each method since it is a standard evaluation metric in Information Retrieval [69]. MAP is calculated as follows:

$$MAP = \frac{1}{|C|}\sum_{c=1}^{|C|}\frac{1}{5}\sum_{k=1}^{5}\frac{1}{k}\sum_{i}^{k} rel(i)$$

where $rel(i)$ is 1 if the URIs have the sub-property relation and 0 otherwise, C is 153 which is the number of the unique Super-property URIs. Moreover, k is the number of pairs with the highest relevant score. For these experiments, k is defined to be 5 since the top 5 Property URIs are kept to generate the candidate pairs along with a Super-property URI as explained in Candidate pairs generation section.

More analytically, for each pre-trained embedding model 9 machine learning algorithms are trained. This means that 36 different models are obtained and compared in terms of the metrics mentioned above. The same data preprocessing pipeline is used for all 36 models. Since each embedding model produces a different vector representation regarding the size, the models may differ in number of features. This happens because the values

of each vector representation are used to produce features. With this assumption the size of each dataset in terms of features is the following: *Sentence-Bert 3094, spaCy 406, Word2vec 1222 and Glove 1222* features. Each algorithm is trained on the training set and tested on the test and candidates set. The best models are furtherly explored by applying techniques like Hyperparameter Optimization and Principal Component analysis. Finally, a presentation of Ontology Scout Tool prototype takes place along with examples regarding retrieving information from an ontology set.

4.1 Implementation

4.2 Sub-Property Link Prediction in European Union's Ontology Set

4.2.1 Results on Test Set

Firstly, the results obtained from the test set are demonstrated. As mentioned above, 36 different models are trained and tested in terms of accuracy and precision. Table 1 provides a table holding the accuracies for each model.

Table 1: Accuracy Scores on Test Set

	Sentence-Bert	**Spacy**	**Word2Vec**	**Glove**
AdaBoost	0,9554	0,9531	0,9554	0,9576
Decision Trees	0,9531	0,9397	0,9509	0,9286
K-NN	0,6674	0,6741	0,9330	0,8348
MLP	0,8839	0,9129	0,9621	0,9598
Naïve Bayes	0,6429	0,8527	0,7121	0,6920
Random Forest	0,9598	0,9576	0,9554	0,9621
SVM	0,9241	0,9754	0,9442	0,9420

SVM Bagging	0,9665	0,9509	0,9643	0,9688
XGBoost	0,9598	0,9621	0,9688	0,9621

As it shown, the SVM in combination with spaCy pre-trained embeddings scores the highest accuracy of 97,54% while SVM Bagging-Glove and XGBoost-Word2Vec come second with 96,88% accuracy. The SVM-Bagging method belongs in 3 out of the top 5 models in terms of accuracy. On the other hand, the Naive Bayes-Sentence-Bert and k-NN-Sentence-Bert models hold the lowest scores in terms of accuracy. In detail, their scores are 64,29% and 66,74 respectively. Figure 13 demonstrates the models in terms of accuracy. Overall, 31 out of 36 models hold scores above 80%. Moreover, the Naive Bayes and k-NN models hold the lowest scores in combination with all 4 pre-trained embedding models.

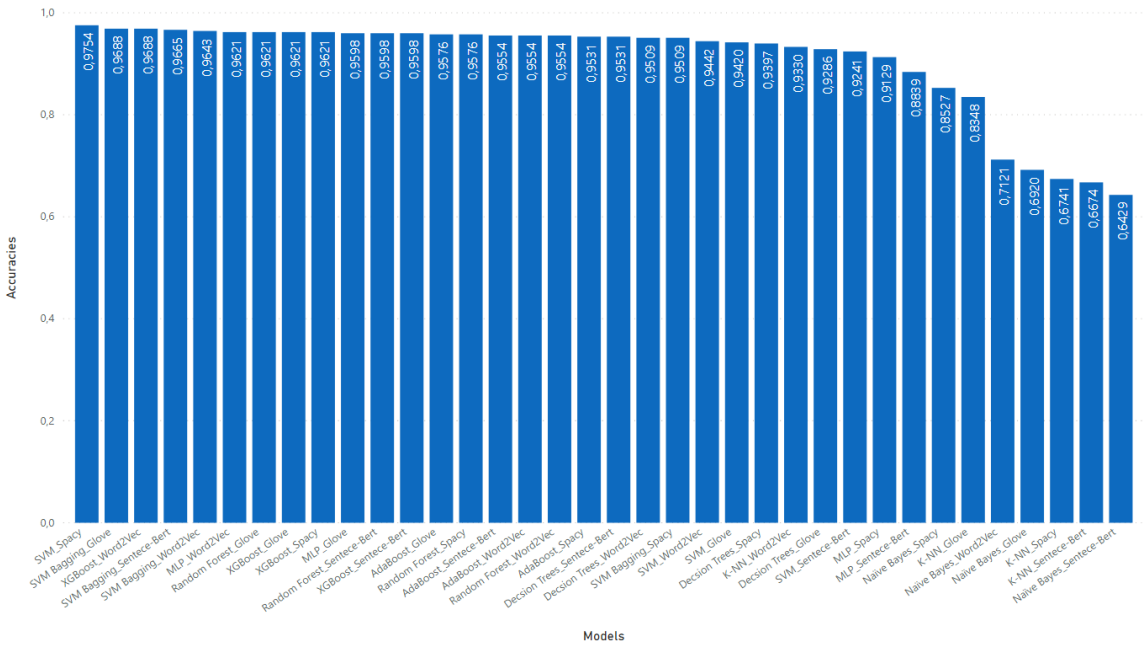


Figure 13: This bar chart demonstrates the combinations of machine learning models and pre-trained embedding models with their accuracies on the Test Set

Regarding the precision of the models, the SVM-spaCy holds the highest value of 97,55%. As before, the SVM Bagging-Glove and XGBoost-Word2Vec models come

after the SVM-spaCy in terms of precision with 96,89% and 96,93 respectively. Oppositely, the k-NN-Sentence-Bert model has the lowest precision of 67,66%. The Naive-Bayes and k-NN models hold the lowest precision scores which aligns with the findings obtained regarding the accuracy of these models. Figure 14 illustrates the relation of accuracy and precision of the models. It demonstrates a strong correlation between the two metrics especially, for accuracies higher than 80%.

Table 2: Precision Scores on Test Set

	Sentence-Bert	**Spacy**	**Word2Vec**	**Glove**
AdaBoost	0,9555	0,9534	0,9557	0,9587
Decision Trees	0,9531	0,9400	0,9512	0,9287
K-NN	0,6766	0,6742	0,9333	0,8365
MLP	0,8842	0,9153	0,9621	0,9600
Naïve Bayes	0,6883	0,8537	0,7388	0,7234
Random Forest	0,9599	0,9578	0,9555	0,9623
SVM	0,9242	0,9755	0,9444	0,9423
SVM Bagging	0,9668	0,9518	0,9649	0,9689
XGBoost	0,9600	0,9625	0,9693	0,9628

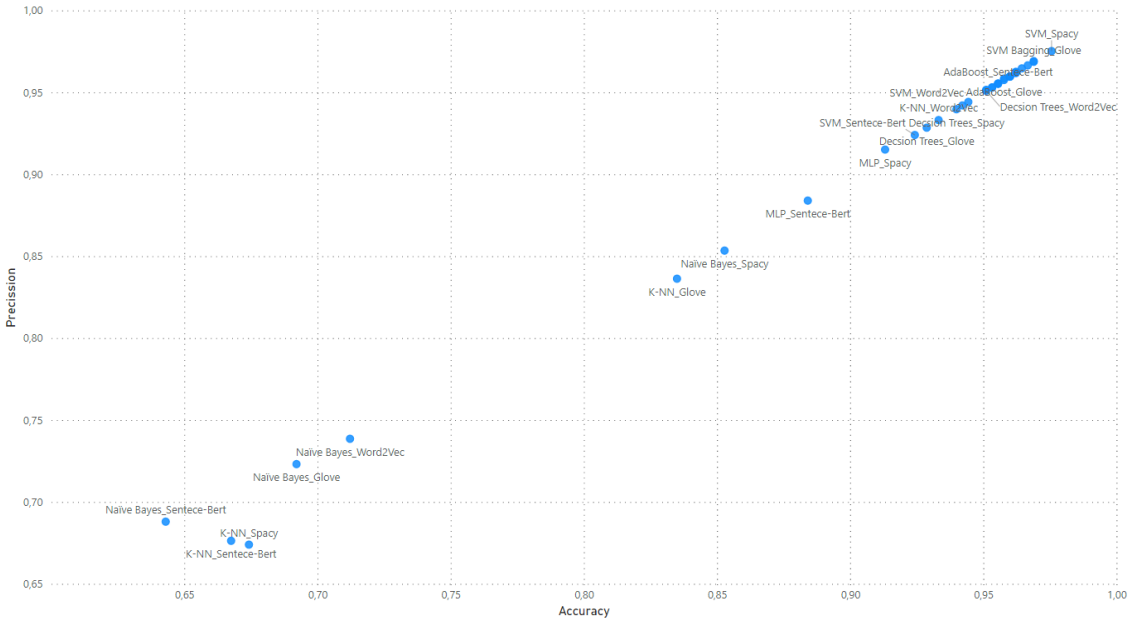


Figure 14: Scatter-plot of the 36 models regarding their precision and accuracy on the Test Set

4.2.2 Results on Candidates Set

In this section, the results obtained from the candidates set are illustrated. The procedure of obtaining the candidate set is thoroughly described in the Dataset section.

Table 3: Accuracy Scores on Candidates Set

	Sentence-Bert	**Spacy**	**Word2Vec**	**Glove**
AdaBoost	0,8392	0,8484	0,8654	0,8484
Decision Trees	0,8588	0,8353	0,8458	0,8549
K-NN	0,6235	0,6824	0,7817	0,7752
MLP	0,8471	0,8340	0,8627	0,8863
Naïve Bayes	0,8340	0,8288	0,8261	0,8340
Random Forest	0,8000	0,8222	0,8183	0,8144
SVM	0,8444	0,8732	0,8680	0,8719

SVM Bagging	0,8654	0,8732	0,8719	0,8680
XGBoost	0,8627	0,8549	0,8680	0,8601

In general, 33 out of the 36 models have an accuracy score between 80% to 88,63%. The MLP-Glove model has the highest value of 88,63%. In addition, the SVM-spaCy and SVM Bagging-spaCy models score 87.32% which is the second highest score. It is with mentioning that the Support Vector Machines models which include the SVM and SVM Bagging methods are found in 7 out of top 10 models as the Figure 15 suggests. Furthermore, all the k-NN models score the lowest accuracy. Similar results found in the test set. Particularly, for the candidates set all the Random-Forests models score slightly higher than the k-NN models. The spaCy pre-trained embedding model holds the 2 out of top 5 models while the Sentence-Bert models holds 2 out of the lowest 5 five models.

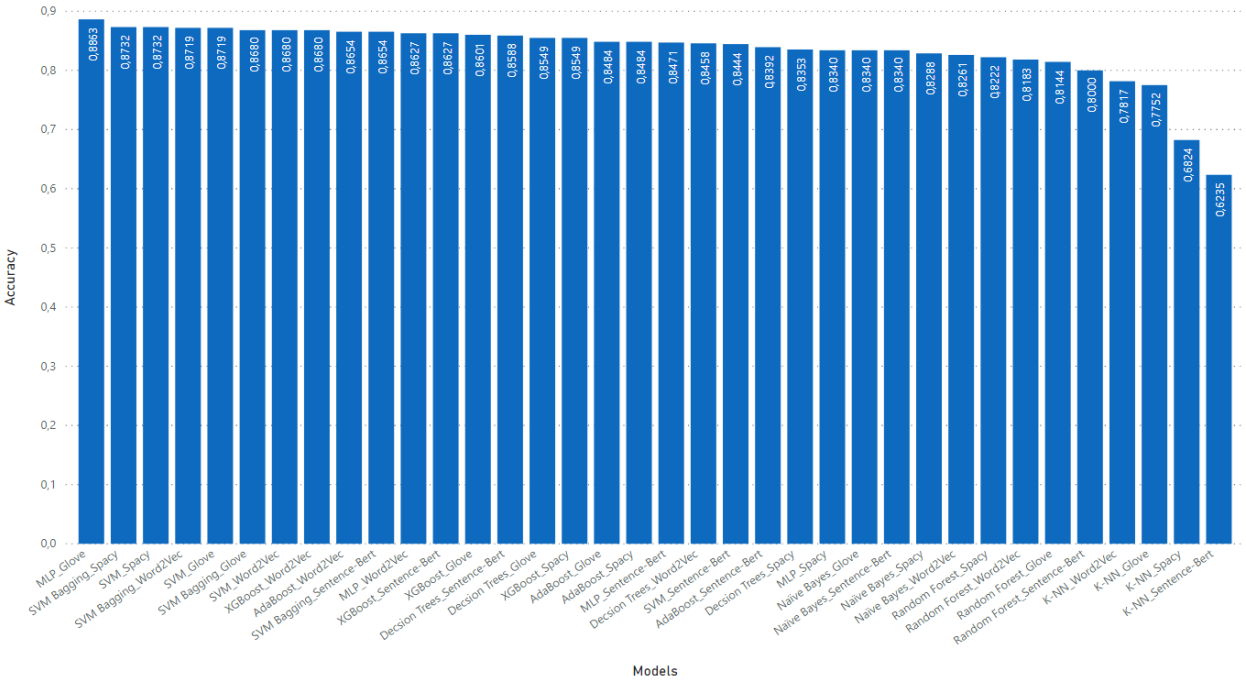


Figure 15: This bar chart demonstrates the combinations of machine learning models and pre-trained embedding models with their accuracies on the Candidates Set

In terms of precision, the SVM-spaCy scores 94,07% which is slightly above the MLP-Glove model's precision of 94,06%. While there is no significant change in the ranking of models regarding precision in comparison with accuracy, the Decision Trees-Sentence-Bert model scores 93.79% in terms of accuracy. On the opposite side, the k-NN-spaCy model holds the lowest precision score of 86.84%.

Table 4: Precision Scores on Candidates Set

	Sentence-Bert	**Spacy**	**Word2Vec**	**Glove**
AdaBoost	0,9294	0,9310	0,9344	0,9310
Decision Trees	0,9379	0,9221	0,9306	0,9291
K-NN	0,8811	0,8684	0,9231	0,9114
MLP	0,9308	0,9286	0,9354	0,9406
Naïve Bayes	0,8979	0,9311	0,9106	0,9073
Random Forest	0,9290	0,9267	0,9278	0,9255
SVM	0,9353	0,9407	0,9349	0,9373
SVM Bagging	0,9391	0,9345	0,9342	0,9381
XGBoost	0,9338	0,9323	0,9349	0,9333

Apart from the accuracy and precision metrics, the “Mean Average Precision” is also measured since it is a more comprehensive metric for evaluating the machine learning models for this problem.

Table 5: Mean Average Precision on Candidates Set

	Sentence-Bert	**Spacy**	**Word2Vec**	**Glove**
AdaBoost	0,1678	0,1697	0,1731	0,1697
Decision Trees	0,1718	0,1671	0,1692	0,1710
K-NN	0,1247	0,1365	0,1563	0,1550
MLP	0,1694	0,1668	0,1725	0,1773
Naïve Bayes	0,1668	0,1658	0,1652	0,1668
Random Forest	0,1600	0,1644	0,1637	0,1629
SVM	0,1689	0,1746	0,1736	0,1744
SVM Bagging	0,1731	0,1746	0,1744	0,1736
XGBoost	0,1725	0,1710	0,1736	0,1720

As illustrated in Table 5 the MLP-Glove along with the SVM Bagging-spaCy and the SVM-spaCy perform better in terms of MAP in comparison with the other models. In detail, the MLP-Glove holds a MAP of 17,73% while the SVM Bagging-spaCy and SVM-spaCy models score 17.46%. Contrarily, the k-NN and Random Forests models hold the lowest MAPs. In depth, MAP ranking aligns with the accuracy ranking for the candidates set. It is found that the MAP and accuracy have a perfect linear relationship with a correlation coefficient equal to 1 as it is demonstrated in Figure 16.

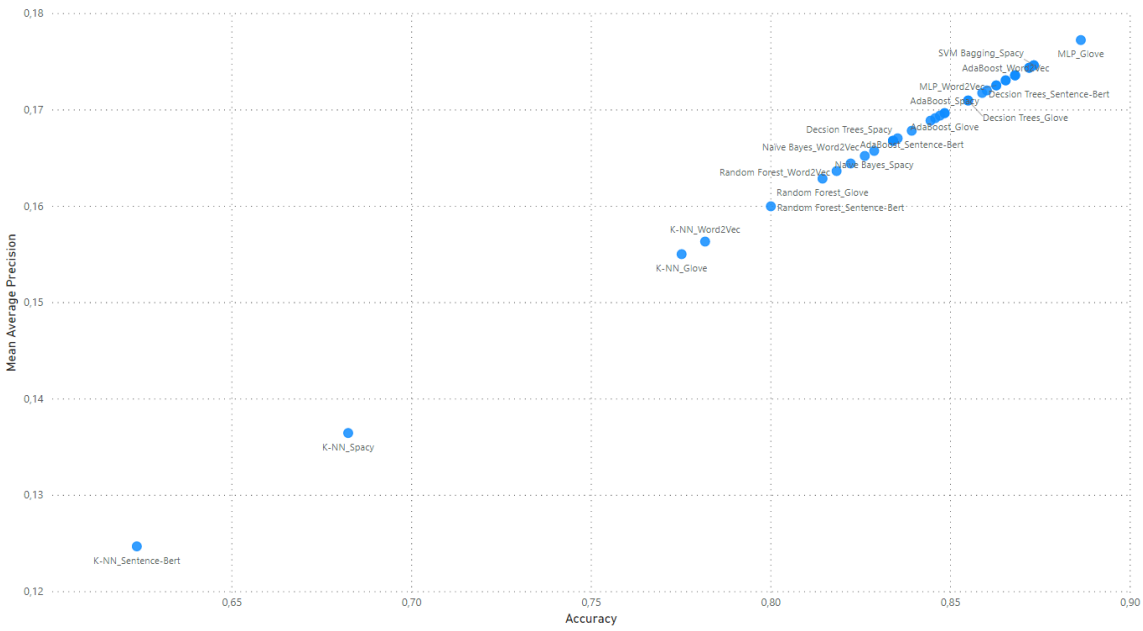


Figure 16: Scatter-plot of the 36 models regarding their mean average precision and accuracy on the Candidates Set

While the MAP and accuracy have a perfect linear relation, the MAP and precision have a correlation equal to 86,04%.

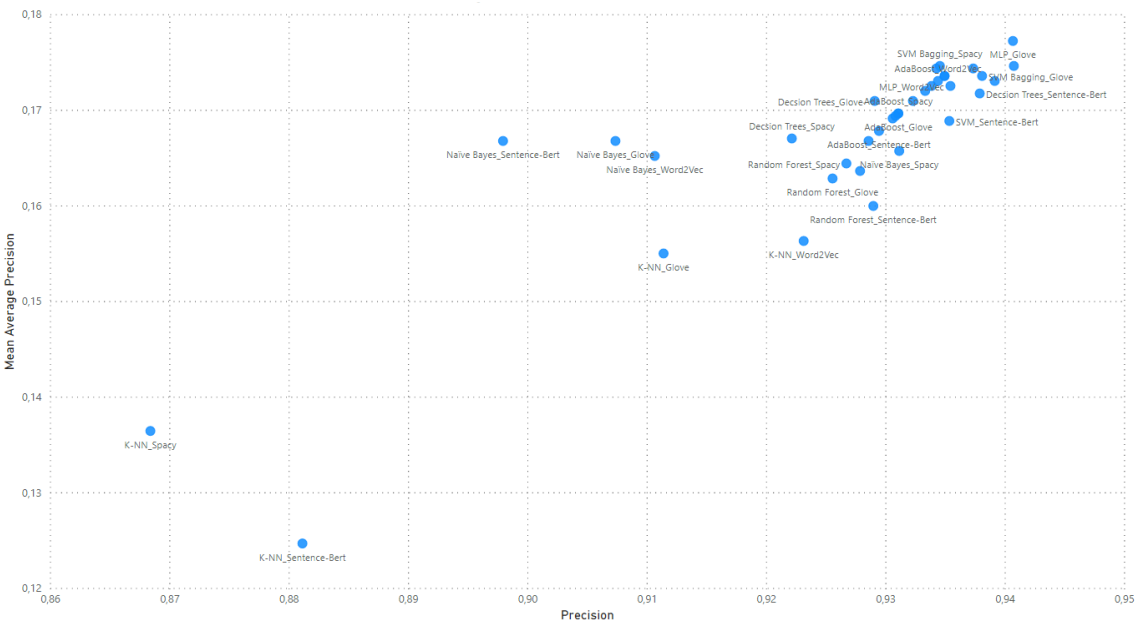


Figure 17: Scatter-plot of the 36 models regarding their mean average precision and precision on the Candidates Set

Overall, the MLP-Glove and SVM-spaCy models outstand the other models in terms of accuracy, precision and mean-average-precision in the candidates set and test set respectively. For this reason, these particular models are used in a Hyperparameter optimization process.

4.2.3 Hyperparameter Optimization

As mentioned above, for the MLP-Glove and SVM-spaCy models are used in a hyperparameter optimization process. The GridSearchCV[20] method is used for applying this process. Specifically, this method is an exhaustive search over pre-specified parameters for a model. It is applied by performing a cross validation to the training set. Then after comparing the models of each optimization regarding an evaluation metric, it returns a set including the best parameters. For the purpose of this thesis, the cross-validation parameter is set to 10 folds while for accuracy is set a evaluation metric. The accuracy metric is preferred over other metrics since it holds a perfect linear relation to the MAP according to the experiments above. Table 6 include the parameters utilized for the optimization process.

[20] https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html

Table 6: Hyperparameter Optimization for MLP-Glove and SVM-spaCy models

MLP-Glove

Parameters	Values
Hidden layer size	5, 10, 15, 20, 25, 30
Batch size	50, 100, 150, 200, 300
Maximum iteration	10, 30, 50, 70

SVM-spaCy

Parameters	Values
Regularization	1, 10
Kernel	Linear, RBF
Gamma	Scale, Auto
Degree	1, 2, 3, 4, 5
Probability	True, False
Maximum iteration	10, 30, 50, 70

Regarding the MLP-Glove optimization, after 1200 different fits of the model the highest accuracy obtained on the training set, using a 10-fold cross validation, is 97.26%. This score is obtained by using 30 hidden layers, 50 as the batch size and 70 maximum iterations. While for the SVM-spaCy the best parameters after 3200 fits are the following: Regularization: 10, Degree: 1, Gamma: Scale, Kernel: RBF, Maximum iteration: 70 and Probability: True. Using these parameters an accuracy of 92,06% is obtained in the training set with a 10-fold cross validation. Since the primary focus of this thesis is to create a machine learning model in predicting the relation among ontology properties in unseen data, the best parameters for each model are used for training the MLP-Glove and SVM-spaCy models. The models are tested on the candidates set using the accuracy, precision and MAP evaluation metrics.

Despite the highest accuracy of the MLP-Glove in the training set during Hyperparameter optimization, the SVM-spaCy obtains a higher score in the candidates set of 90,2%. Moreover, the MLP-Glove has a precision of 93,62% while the SVM-spaCy 92,24%. In terms of the MAP the SVM-spaCy outperforms MLP-Glove with a value of 18,03%.

Embracing the results for the candidates set before the Hyperparameter optimization, for MLP-Glove there is slight improvement of 0,44% in precision after optimization while the accuracy and MAP are decreasing. In addition, for the SVM-spaCy model the accuracy and MAP are increased to 90,2% and 18.03% respectively. Overall, these are the highest scores obtained for these evaluation metrics regarding candidates set so far. On the other hand, there is a decrease in the precision of the SVM-spaCy model after optimization.

Table 7: Metrics before and after the Hyperparameter Optimization Process

	Accuracy		**Precision**		**MAP**	
	Before Optimization	**After Optimization**	**Before Optimization**	**After Optimization**	**Before Optimization**	**After Optimization**
MLP-Glove	0,8863	0.8588	0,9406	0.9362	0,1773	0.1717
SVM-spaCy	0,8732	0.902	0,9353	0.9222	0,1746	0.1803

4.2.4 Principal Component Analysis

Considering the evaluation metrics obtained for both models regarding the candidates set after the Hyperparameter optimization, the SVM-spaCy model along with its best parameters is used in Principal Component Analysis (PCA) implementation since it holds the highest accuracy and MAP obtained in the experiments. PCA aims to determine the optimal number of components for a machine learning model which maximizes its performance. As mentioned in previous sections, the SVM-spaCy model is trained on 406 features. Since the candidates set is considered to be unseen data, PCA is performed on the training set using cross validation. In order to decrease the computational cost, a 5-fold cross validation takes place. Moreover, the evaluation metric regarding the performance is set to be the accuracy because of its linear relation with the MAP. Finally, in contemplation of the computational cost, PCA is performed in the range of 400 to 10 components decreasing by 10. As demonstrated in Figure 18, the highest accuracy is scored using 170 components.

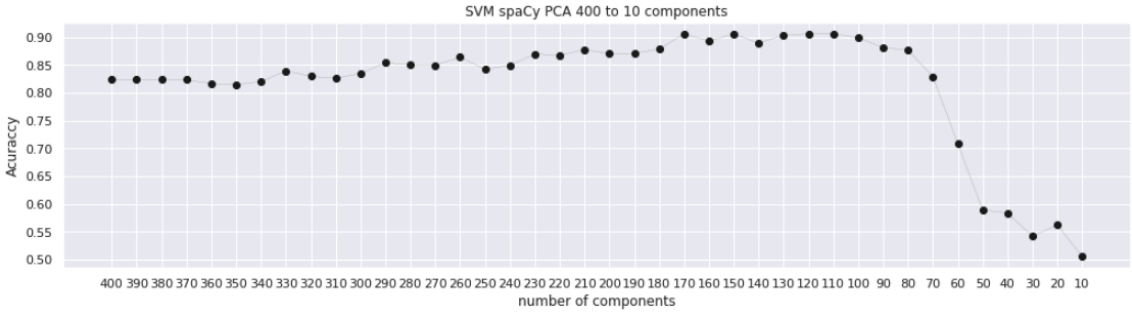


Figure 18: Principle Component Analysis SVM-spaCy from 400 to 10 components decreasing by 10 components

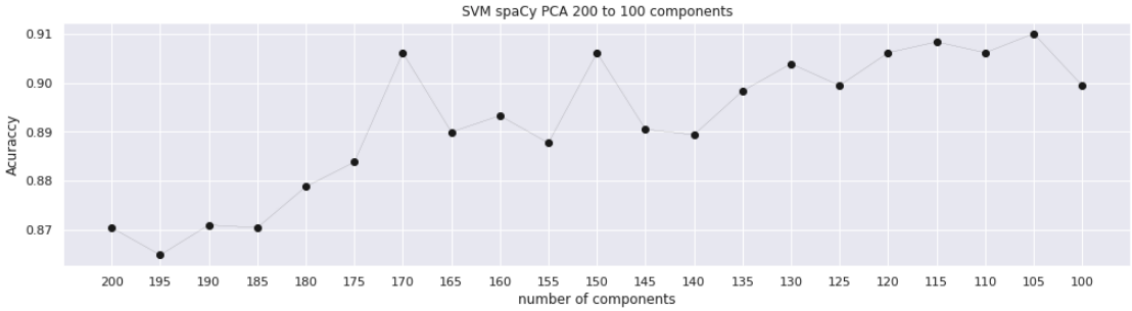


Figure 19: Principle Component Analysis SVM-spaCy from 200 to 100 components decreasing by 5 components

Due to this fact, PCA is applied in the range of 200 to 100 decreasing by 10 components. In conclusion, since the accuracy is increasing after 160 components and starts decreasing after 120, PCA takes place in the range of 170 to 101 decreasing by 1 component. According to the score obtained from the training set, using 5-fold cross validation, the optimal number of components is 141 with 91,56% of accuracy.

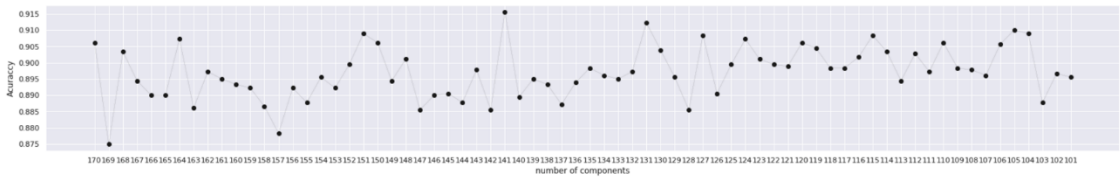


Figure 20: Principle Component Analysis SVM-spaCy from 170 to 101 components decreasing by 1 component

Finally, the SVM-spaCy model is trained using the best parameters and PCA is applied using 141 components and tested on the candidates set. A significant decrease in accuracy and the MAP evaluation metrics is observed after PCA. While the precision is decreased to 91,5%, the accuracy and MAP scores are 72,02% and 14,4% respectively as it is demonstrated in Table 8.

Table 8: Metrics before and after Principal Component Analysis

	Accuracy		**Precision**		**MAP**	
	Before PCA	**After PCA**	**Before PCA**	**After PCA**	**Before PCA**	**After PCA**
SVM-spaCy	0.902	0.7202	0.9222	0.915	0.1803	0.144

4.3 Ontology Scout Tool (OS Tool)

OS Tool is an ontology search tool based on vector representations. Its aim is to help ontology engineers during ontology development and ontology alignment tasks. It is built using SBERT pre-trained embeddings. Regarding its graphical user interface, the Tkinter python's library[21] is utilized. Users provide a search query and retrieve relevant entities of the ontologies which are pre-loaded to the OS Tool. For the purpose of this thesis, the European Union's ontologies are pre-loaded creating an ontology set as explained in the Ontology parsing section.

The Figure 21 demonstrates a prototype of the OS Tool. As mentioned above, users provide the tool with a query. The query may contain multiple keywords in the form of a phrase. Then this query is transformed to an embedding using SBERT. The query embedding is placed in the vector space among with the other embeddings generated from the labels and comments of the ontology set. After that, the query embedding is compared with all other embeddings in terms of cosine similarity. Since an entity may have a label

[21] https://docs.python.org/3/library/tkinter.html

and a comment the maximum cosine similarity between their embeddings and the query embedding is considered as the similarity of the query and the entity. This means that if the cosine similarity of the query-label embeddings is greater than the query-comment embeddings then the first one is set as the similarity between the query and the entity. Finally, a sorted list of the entities is retrieved according to the similarity with the query.

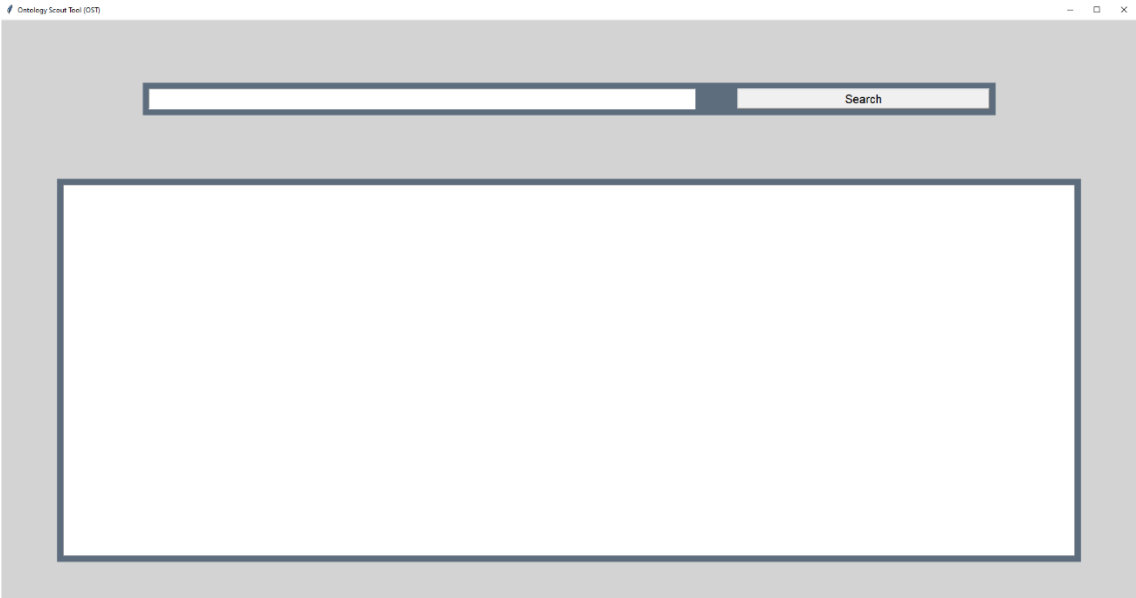
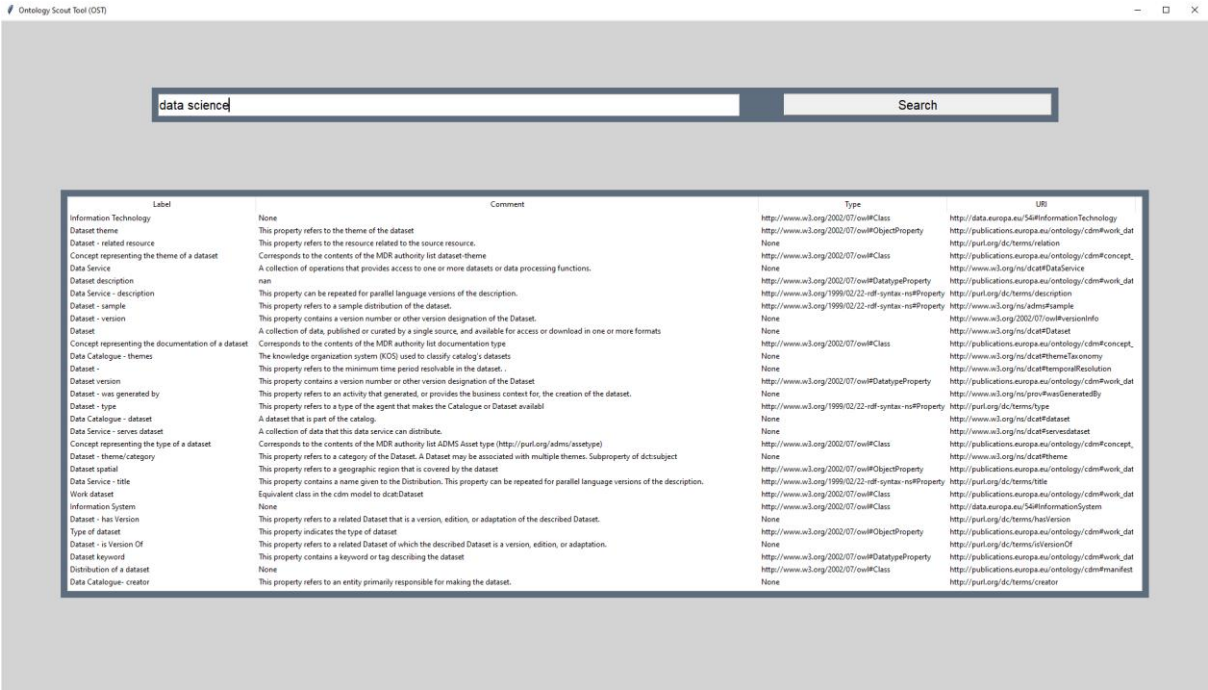


Figure 21: The Ontology Scout Tool GUI

Assuming that users provide the query “data science”, a table of the entities is retrieved descending on the cosine similarity which is obtained as explained above. The table of the entities contains information regarding their label, comment, type and the URI.



Label	Comment	Type	URI
Information Technology	None	http://www.w3.org/2002/07/owl#Class	http://data.europa.eu/54i#InformationTechnology
Dataset theme	This property refers to the theme of the dataset	http://www.w3.org/2002/07/owl#ObjectProperty	http://publications.europa.eu/ontology/cdm#work_dat
Dataset - related resource	This property refers to the resource related to the source resource.	None	http://purl.org/dc/terms/relation
Concept representing the theme of a dataset	Corresponds to the contents of the MDR authority list dataset-theme	http://www.w3.org/2002/07/owl#Class	http://publications.europa.eu/ontology/cdm#concept_
Data Service	A collection of operations that provides access to one or more datasets or data processing functions.	None	http://www.w3.org/ns/dcat#DataService
Dataset description	nan	http://www.w3.org/2002/07/owl#DatatypeProperty	http://publications.europa.eu/ontology/cdm#work_dat
Data Service - description	This property can be repeated for parallel language versions of the description.	http://www.w3.org/1999/02/22-rdf-syntax-ns#Property	http://purl.org/dc/terms/description
Dataset - sample	This property refers to a sample distribution of the dataset.	http://www.w3.org/1999/02/22-rdf-syntax-ns#Property	http://www.w3.org/ns/adms#sample
Dataset - version	This property contains a version number or other version designation of the Dataset.	None	http://www.w3.org/2002/07/owl#versionInfo
Dataset	A collection of data, published or curated by a single source, and available for access or download in one or more formats	None	http://www.w3.org/ns/dcat#Dataset
Concept representing the documentation of a dataset	Corresponds to the contents of the MDR authority list documentation type	http://www.w3.org/2002/07/owl#Class	http://publications.europa.eu/ontology/cdm#concept_
Data Catalogue - themes	The knowledge organization system (KOS) used to classify catalog's datasets	None	http://www.w3.org/ns/dcat#ThemeTaxonomy
Dataset -	This property refers to the minimum time period resolvable in the dataset. .	None	http://www.w3.org/ns/dcat#temporalResolution
Dataset version	This property contains a version number or other version designation of the Dataset	http://www.w3.org/2002/07/owl#DatatypeProperty	http://publications.europa.eu/ontology/cdm#work_dat
Dataset - was generated by	This property refers to an activity that generated, or provides the business context for, the creation of the dataset.	None	http://www.w3.org/ns/prov#wasGeneratedBy
Dataset - type	This property refers to a type of the agent that makes the Catalogue or Dataset availabl	http://www.w3.org/1999/02/22-rdf-syntax-ns#Property	http://purl.org/dc/terms/type
Data Catalogue - dataset	A dataset that is part of the catalog.	None	http://www.w3.org/ns/dcat#dataset
Data Service - serves dataset	A collection of data that this data service can distribute.	None	http://www.w3.org/ns/dcat#servesdataset
Concept representing the type of a dataset	Corresponds to the contents of the MDR authority list ADMS Asset type (http://purl.org/adms/assetype)	http://www.w3.org/2002/07/owl#Class	http://publications.europa.eu/ontology/cdm#concept_
Dataset - theme/category	This property refers to a category of the Dataset. A Dataset may be associated with multiple themes. Subproperty of dct:subject	None	http://www.w3.org/ns/dcat#theme
Dataset spatial	This property refers to a geographic region that is covered by the dataset	http://www.w3.org/2002/07/owl#ObjectProperty	http://publications.europa.eu/ontology/cdm#work_dat
Data Service - title	This property contains a name given to the Distribution. This property can be repeated for parallel language versions of the description.	http://www.w3.org/1999/02/22-rdf-syntax-ns#Property	http://purl.org/dc/terms/title
Work dataset	Equivalent class in the cdm model to dcat:Dataset	http://www.w3.org/2002/07/owl#Class	http://publications.europa.eu/ontology/cdm#work_dat
Information System	None	http://www.w3.org/2002/07/owl#Class	http://data.europa.eu/54i#InformationSystem
Dataset - has Version	This property refers to a related Dataset that is a version, edition, or adaptation of the described Dataset.	None	http://purl.org/dc/terms/hasVersion
Type of dataset	This property indicates the type of dataset	http://www.w3.org/2002/07/owl#ObjectProperty	http://publications.europa.eu/ontology/cdm#work_dat
Dataset - is Version Of	This property refers to a related Dataset of which the described Dataset is a version, edition, or adaptation.	None	http://purl.org/dc/terms/isVersionOf
Dataset keyword	This property contains a keyword or tag describing the dataset	http://www.w3.org/2002/07/owl#DatatypeProperty	http://publications.europa.eu/ontology/cdm#work_dat
Distribution of a dataset	None	http://www.w3.org/2002/07/owl#Class	http://publications.europa.eu/ontology/cdm#manifest
Data Catalogue- creator	This property refers to an entity primarily responsible for making the dataset.	None	http://purl.org/dc/terms/creator

Figure 22: The results of “data science” query on the Ontology Scout Tool

5 Discussion

In the previous section, several experiments are conducted in order to predict the relation among properties in an ontology set. As the results indicate, the Support Vector Machines along with the spaCy pre-trained embeddings are the best combination of models in the problem of this thesis. In detail, the SVM-spaCy model holds the first place in terms of accuracy and precision in the test set and in precision regarding the candidates set. Furthermore, it retains the second place in terms of accuracy and MAP in the candidates set having slightly lower values than the Multilayer Perceptron-Glove model. After the hyperparameter optimization at the SVM-spaCy model there is an increase in the accuracy and MAP while the precision decreases imperceptibly. In addition, applying Principal Component Analysis results to a higher accuracy in the training set but a notable decrease in the candidates set. A justification of the lower accuracy after PCA may be the loss of information by shrinking the dimensions of the dataset. Taking into account the overall performance of the models applied for this problem, the majority of models has an accuracy above 80% in the candidates set. This result is encouraging since it justifies that the machine learning models are able to identify the pairs which have the sub-property relation despite the fact their labels and comments are related according to their embeddings. Specifically, all algorithms except the k-NN models score above the threshold of 80%.

It is worth mentioning that the Support Vector Machine techniques (SVM and SVM Bagging) have a significant high performance regarding this problem both in the test set and the candidates set. The candidates set is generated using the Sentence-Bert pre-trained embeddings. SBERT produces sentence embeddings of the size of 768 dimensions. Having such a number of dimensions provides a great detail in the vector space. Despite this fact, SBERT in combination with the machine learning algorithms has a relatively low ranking in comparison with the other pre-trained embedding models. While the results are promising, there are some limitations which this thesis is aware of. First of all, the size of the dataset is relatively small since it focuses on European Union's vocabularies which hold the sub-property relation. Regarding this relation, it denotes the relation

among a property and its properties belonging under the first one rather than the fact that two properties are the same.

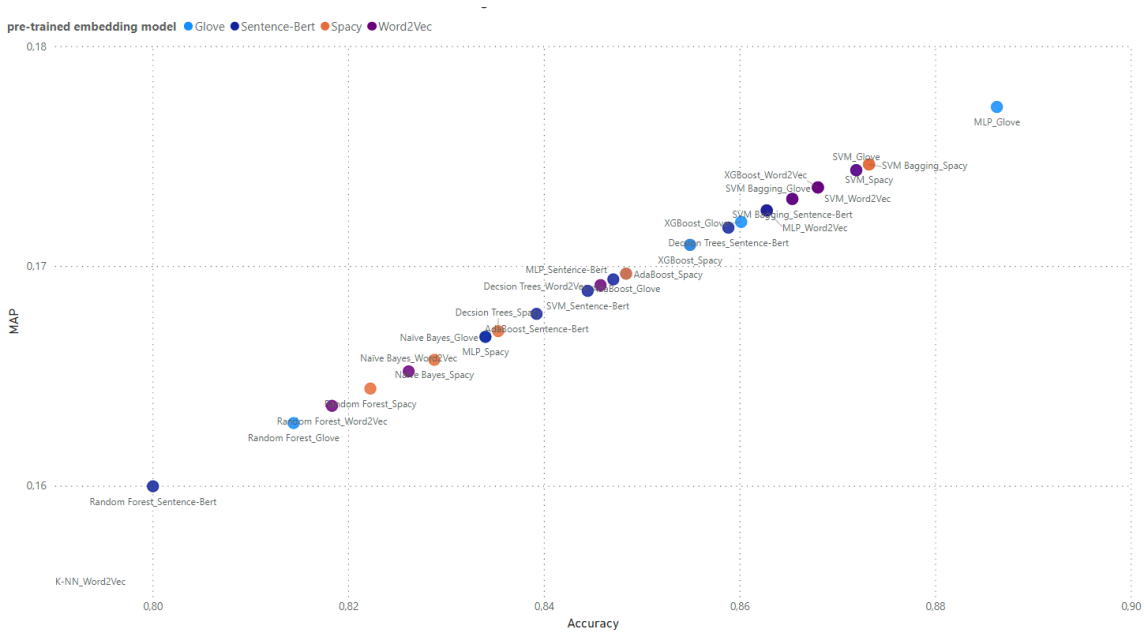


Figure 23: Scatter-plot of the 36 models regarding their mean average precision and accuracy on the Candidates Set before Hyperparameter Optimization and PCA. The colors indicate the pre-trained embedding model.

Apart from the sub-property link prediction problem, an ontology search tool is introduced named Ontology Scout Tool (OS Tool). The prototype created provides ontology engineers with information regarding a query. This tool may support their decisions regarding ontology development and alignment tasks. It is based on SBERT pre-trained embeddings model because it is fast and reliable in computing the query embedding. Ontologies aim to propel data integration and knowledge extraction via reusing concepts and relations that already exist. Using this tool, ontology engineers may acquire a strong indication about which ontologies have an association with their query helping them to target specific ontologies and explore them in detail. One of the great advantages of the OS Tool is that ontology engineers can pre-load the ontologies of their interest. As stated in previous sections, the OS Tool is a prototype and hence it is an early stage. This means that there are several further steps that can improve the OS Tool. The results retrieved from the OS Tool rely on the SBERT embeddings which rely on a certain methodology and a specific corpus. For this reason, several pre-trained embedding models should be

tested in regards to the OS Tool. A more sophisticated way may be the average cosine similarity among the entities and the query obtained by several pre-trained embedding models. In addition, other distance metrics may be used to measure the similarity between the query and the entities in the vector space. Furthermore, at this early stage the OS Tool takes into consideration only the labels and comments of an URI rather than its relations with other entities. Finally, further improvements may take place regarding the graphical user interface of the tool but for the purpose of this thesis they are out of scope.

6 Conclusion and Future Work

This thesis tried to improve ontology engineering by implementing machine learning techniques and pre-trained vector representation models in an ontology set. Specifically, the ontologies used are concerning the domain of e-Government in European Union. These ontologies are retrieved from the official site of the European Union's Publication office. More thoroughly, this thesis tackled the problem of sub-property link prediction in an ontology set while introducing an ontology search tool that aims to assist ontology engineers in their tasks. The results proved that predicting the sub-property relation is feasible with a significant high accuracy. Furthermore, the ontology search tool helps ontology engineers to focus on their concepts and ontologies of interest by providing a query.

Support Vector Machines (SVM) in combination with the "en_core_web_sm" spaCy pre-trained embedding model outperformed the rest of the models in the problem of sub-property link prediction. After the parameter optimization, the accuracy obtained from the SVM-spaCy model is 90.2%. Furthermore, the ontology search tool which is based on Sentence-Bert pre-trained embedding model can help ontology engineers to easily identify existing ontologies related to the entities of their interest. The results of this thesis indicate that applying machine learning techniques in the field of ontology engineering is attainable and can help reduce the extensive human labor.

Since the dataset generated was based on European Union's ontologies, a possible future direction is to apply these methodologies in various sets of ontologies. Also, the problem of link prediction focused on the sub-property relation among properties. Another future direction is to focus on the sub-class relation among entities in an ontology set. Finally, regarding the ontology search tool, the vector representations are obtained from SBERT pre-trained embedding model which relies on a specific corpus. As a next step, the pre-trained embeddings can be generated from a corpus from a domain which aligns with the ontologies of users' interest.

Bibliography

[1]. Berners-Lee, T., Hendler, J., & Lassila, O. (2001). The semantic web. *Scientific american*, *284*(5), 34-43.

[2]. Feigenbaum, L., Herman, I., Hongsermeier, T., Neumann, E., & Stephens, S. (2007). The semantic web in action. *Scientific American*, *297*(6), 90-97.

[3]. Klischewski, R. (2003, September). Semantic web for e-government. In *International Conference on Electronic Government* (pp. 288-295). Springer, Berlin, Heidelberg.

[4]. Kendall, E. F., & McGuinness, D. L. (2019). Ontology engineering. *Synthesis Lectures on The Semantic Web: Theory and Technology*, *9*(1), i-102.

[5]. Gal, A. (2009). Ontology Engineering*. In Encyclopedia of Database Systems* (pp.1972-1973). Springer US

[6]. Vandenbussche, P. Y., Atemezing, G. A., Poveda-Villalón, M., & Vatant, B. (2017). Linked Open Vocabularies (LOV): a gateway to reusable semantic vocabularies on the Web. *Semantic Web*, *8*(3), 437-452.

[7]. Goldberg, Y. (2017). Neural network methods for natural language processing. *Synthesis Lectures on Human Language Technologies*, *10*(1), 1-309.

[8]. Harris, Z. S. (1954). Distributional structure. *Word*, *10*(2-3), 146-162.

[9]. Naili, M., Chaibi, A. H., & Ghezala, H. H. B. (2017). Comparative study of word embedding methods in topic segmentation. *Procedia computer science*, *112*, 340-349.

[10]. Li, Y., & Yang, T. (2018). Word embedding for understanding natural language: a survey. In *Guide to Big Data Applications* (pp. 83-104). Springer, Cham.

[11]. Rajaraman, A., & Ullman, J. D. (2011). *Mining of massive datasets*. Cambridge University Press.

[12]. Chen, X., Chen, B., Zhang, C., & Hao, T. (2017, September). Discovering the recent research in natural language processing field based on a statistical approach. In *International Symposium on Emerging Technologies for Education* (pp. 507-517). Springer, Cham.

[13]. Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems* (pp. 3111-3119).

[14]. Pennington, J., Socher, R., & Manning, C. D. (2014, October). Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)* (pp. 1532-1543).

[15]. Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781.*

[16]. Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. (2017). Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, *5*, 135-146.

[17]. Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., & Zettlemoyer, L. (2018). Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*.

[18]. Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

[19]. Pan, S. J., & Yang, Q. (2009). A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, *22*(10), 1345-1359.

[20]. Jain, S., & Kumar, P. Semantic Web, Ontologies and E-Government: A Review.

[21]. Zhang, Y., Wang, X., Lai, S., He, S., Liu, K., Zhao, J., & Lv, X. (2014). Ontology matching with word embeddings. In *Chinese computational linguistics and natural language processing based on naturally annotated big data* (pp. 34-45). Springer, Cham.

[22]. Li, J., Tang, J., Li, Y., & Luo, Q. (2008). Rimom: A dynamic multistrategy ontology alignment framework. *IEEE Transactions on Knowledge and data Engineering*, *21*(8), 1218-1232.

[23]. Jean-Mary, Y., & Kabuka, M. (2007, September). Asmov: Ontology alignment with semantic validation. In *Joint SWDB-ODBIS Workshop* (pp. 15-20).

[24]. Cruz, I. F., Antonelli, F. P., & Stroe, C. (2009). AgreementMaker: efficient matching for large real-world schemas and ontologies. *Proceedings of the VLDB Endowment*, *2*(2), 1586-1589.

[25]. Pembeci, İ. (2016). Using word embeddings for ontology enrichment. *International Journal of Intelligent Systems and Applications in Engineering*, *4*(3), 49-56.

[26]. Gupta, N., Podder, S., Sengupta, S., & Annervaz, K. M. (2016, December). Domain ontology induction using word embeddings. In *2016 15th IEEE International Conference on Machine Learning and Applications (ICMLA)* (pp. 115-119). IEEE.

[27]. Saedi, C., Branco, A., Rodrigues, J., & Silva, J. (2018, July). Wordnet embeddings. In *Proceedings of the third workshop on representation learning for NLP* (pp. 122-131).

[28]. Kolyvakis, P., Kalousis, A., & Kiritsis, D. (2018, June). Deepalignment: Unsupervised ontology matching with refined word vectors. In *Proceedings of the 2018*

Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers) (pp. 787-798).

[29]. Mrkšić, N., Séaghdha, D. O., Thomson, B., Gašić, M., Rojas-Barahona, L., Su, P. H., ... & Young, S. (2016). Counter-fitting word vectors to linguistic constraints. *arXiv preprint arXiv:1603.00892*.

[30]. Gromann, D., & Declerck, T. (2018, May). Comparing pretrained multilingual word embeddings on an ontology alignment task. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

[31]. Al-Rfou, R., Perozzi, B., & Skiena, S. (2013). Polyglot: Distributed word representations for multilingual nlp. *arXiv preprint arXiv:1307.1662*.

[32]. Joulin, A., Grave, E., Bojanowski, P., Douze, M., Jégou, H., & Mikolov, T. (2016). Fasttext. zip: Compressing text classification models. *arXiv preprint arXiv:1612.03651*.

[33]. Meimaris, M. (2019). Visualizing Implicit RDF Schema with the Help of Embeddings. In *EDBT/ICDT Workshops*.

[34]. Ristoski, P., & Paulheim, H. (2016, October). Rdf2vec: Rdf graph embeddings for data mining. In *International Semantic Web Conference* (pp. 498-514). Springer, Cham.

[35]. Jurisch, M., & Igler, B. (2018). RDF2Vec-based classification of ontology alignment changes. *arXiv preprint arXiv:1805.09145*.

[36]. Portisch, J. P. Towards Matching of Domain-Specific Schemas Using General-Purpose External Background Knowledge.

[37]. Portisch, J., & Paulheim, H. (2018). ALOD2Vec matcher. *OM@ ISWC*, *2288*, 132-137.

[38]. Hertling, S., & Paulheim, H. (2017, October). WebIsALOD: providing hypernymy relations extracted from the web as linked open data. In *International Semantic Web Conference* (pp. 111-119). Springer, Cham.

[39]. Wang, M., Wang, R., Liu, J., Chen, Y., Zhang, L., & Qi, G. (2018, October). Towards empty answers in sparql: Approximating querying with rdf embedding. In *International semantic web conference* (pp. 513-529). Springer, Cham.

[40]. Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., & Yakhnenko, O. (2013). Translating embeddings for modeling multi-relational data. In *Advances in neural information processing systems* (pp. 2787-2795).

[41]. Gutiérrez-Basulto, V., & Schockaert, S. (2018). From knowledge graph embedding to ontology embedding? An analysis of the compatibility between vector space representations and rules. *arXiv preprint arXiv:1805.10461*.

[42]. Toutanova, K., Chen, D., Pantel, P., Poon, H., Choudhury, P., & Gamon, M. (2015, September). Representing text for joint embedding of text and knowledge bases.

In *Proceedings of the 2015 conference on empirical methods in natural language processing* (pp. 1499-1509).

[43]. Grover, A., & Leskovec, J. (2016, August). node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 855-864).

[44]. Cochez, M., Ristoski, P., Ponzetto, S. P., & Paulheim, H. (2017, October). Global RDF vector space embeddings. In *International Semantic Web Conference* (pp. 190-207). Springer, Cham.

[45]. Diaz, G. I., Fokoue, A., & Sadoghi, M. (2018). EmbedS: Scalable, Ontology-aware Graph Embeddings. In *EDBT* (pp. 433-436).

[46]. Hao, J., Chen, M., Yu, W., Sun, Y., & Wang, W. (2019, July). Universal representation learning of knowledge bases by jointly embedding instances and ontological concepts. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (pp. 1709-1719).

[47]. Ferré, A., Deléger, L., Zweigenbaum, P., & Nédellec, C. (2018, May). Combining rule-based and embedding-based approaches to normalize textual entities with an ontology. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

[48]. Portisch, J., Hladik, M., & Paulheim, H. (2020). KGvec2go--Knowledge Graph Embeddings as a Service. *arXiv preprint arXiv:2003.05809*.

[49]. Holter, O. M., Myklebust, E. B., Chen, J., & Jimenez-Ruiz, E. (2019). Embedding OWL ontologies with OWL2Vec. In *CEUR Workshop Proceedings* (Vol. 2456, pp. 33-36). Technical University of Aachen.

[50]. Chen, M., Tian, Y., Chen, X., Xue, Z., & Zaniolo, C. (2018, May). On2vec: Embedding-based relation prediction for ontology population. In *Proceedings of the 2018 SIAM International Conference on Data Mining* (pp. 315-323). Society for Industrial and Applied Mathematics.

[51]. Culotta, A., & Sorensen, J. (2004, July). Dependency tree kernels for relation extraction. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)* (pp. 423-429).

[52]. Fundel, K., Küffner, R., & Zimmer, R. (2007). RelEx—Relation extraction using dependency parse trees. *Bioinformatics*, *23*(3), 365-371.

[53]. Giuliano, C., & Gliozzo, A. (2008, August). Instance-based ontology population exploiting named-entity substitution. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)* (pp. 265-272).

[54]. Mousavi, H., Atzori, M., Gao, S., & Zaniolo, C. (2014). Text-mining, structured queries, and knowledge management on web document corpora. *ACM SIGMOD Record*, *43*(3), 48-54.

[55]. Wang, T., Li, Y., Bontcheva, K., Cunningham, H., & Wang, J. (2006, June). Automatic extraction of hierarchical relations from text. In *European Semantic Web Conference* (pp. 215-229). Springer, Berlin, Heidelberg.

[56]. Benarab, A., Rafique, F., & Sun, J. (2019, February). An Ontology Embedding Approach Based on Multiple Neural Networks. In *Proceedings of the 2019 11th International Conference on Machine Learning and Computing* (pp. 186-190).

[57]. Alshargi, F., Shekarpour, S., Soru, T., & Sheth, A. P. (2018). Metrics for evaluating quality of embeddings for ontological concepts.

[58]. Plassard, M. F. (Ed.). (1998). *Functional requirements for bibliographic records* (Vol. 19). De Gruyter Saur.

[59]. Reimers, N., & Gurevych, I. (2019). Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

[60]. Marcus, R. W. E. H. M., Palmer, M., Ramshaw, R. B. S. P. L., & Xue, N. OntoNotes: A Large Training Corpus for Enhanced Processing.

[61]. Altman, N. S. (1992). An introduction to kernel and nearest-neighbor nonparametric regression. *The American Statistician*, *46*(3), 175-185.

[62]. Wu, X., Kumar, V., Quinlan, J. R., Ghosh, J., Yang, Q., Motoda, H., ... & Zhou, Z. H. (2008). Top 10 algorithms in data mining. *Knowledge and information systems*, *14*(1), 1-37.

[63]. Ho, T. K. (1998). The random subspace method for constructing decision forests. *IEEE transactions on pattern analysis and machine intelligence*, *20*(8), 832-844.

[64]. Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine learning*, *20*(3), 273-297.

[65]. Polikar, R. (2006). Ensemble based systems in decision making. *IEEE Circuits and systems magazine*, *6*(3), 21-45.

[66]. Onan, A., Korukoğlu, S., & Bulut, H. (2016). Ensemble of keyword extraction methods and classifiers in text classification. *Expert Systems with Applications*, *57*, 232-247.

[67]. Hastie, T., Rosset, S., Zhu, J., & Zou, H. (2009). Multi-class adaboost. *Statistics and its Interface*, *2*(3), 349-360.

[68]. Chen, T., & Guestrin, C. (2016, August). Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining* (pp. 785-794).

[69]. Beitzel, S. M., Jensen, E. C., Frieder, O., LIU, L., & ÖZSU, M. (2009). MAP.