Johannes Gutenberg University Mainz

JOHANNES GUTENBERG UNIVERSITÄT MAINZ JG|U

Faculty 08: Physics, Mathematics and Computer Science
Institute of Computer Science

Master Thesis

Focusing knowledge-based graph argument mining via topic modeling

Patrick Abels

1. Reviewer Prof. Dr. Stefan Kramer
Institute of Computer Science
Johannes Gutenberg University Mainz

2. Reviewer Jun.-Prof. Dr. Panagiotis Bouros
Institute of Computer Science
Johannes Gutenberg University Mainz

Supervisors Dr. Zahra Ahmadi and Dr. Sophie Burkhardt

27.10.2020

Master Thesis
Title: *Focusing knowledge-based graph argument mining via topic modeling*
Submission date: 27.10.2020
Reviewers: Prof. Dr. Stefan Kramer and Jun.-Prof. Dr. Panagiotis Bouros
Supervisors: Dr. Zahra Ahmadi and Dr. Sophie Burkhardt

This thesis is submitted in partial fulfillment of the requirements for the degree Master of Education (M.Ed.) in Mathematics and Computer Science to:

Johannes Gutenberg University Mainz
Institute of Computer Science
Faculty 08: Physics, Mathematics and Computer Science
Staudingerweg 7
55128 Mainz

Patrick Abels

Abstract

Decision-making usually takes five steps: identifying the problem, collecting data, extracting evidence, identifying pro and con arguments, and making decisions. Focusing on extracting evidence, we study the task of sentence-level argument mining. Given a topic and a sentence, the goal is to classify whether a sentence represents an argument in regard to the topic. As arguments mostly require some degree of world knowledge to be identified and understood, this thesis presents a graph-based model that obtains external knowledge from structured and unstructured data. Because unstructured data generally is not topic specific and noisy, recent work either utilizes small amounts of data or focuses only one type of data. We use a topic model to extract topic- and sentence-specific evidence from the structured knowledge base Wikidata, building a sparse graph based on the cosine similarity between the entity word vectors of Wikidata and the vector of the given sentence. To tackle the general incompleteness of structured knowledge bases, we build a second graph based on topic-specific articles found via Google. Combining these graphs, we successfully capitalize on both structured and unstructured data, achieving 84.16% on accuracy and 65.60% on F1 on the UKP Sentential Argument Mining Corpus.[1][2]

Abstract (German)

Der Prozess der Entscheidungsfindung lässt sich für gewöhnlich in fünf Schritte untergliedern: Identifizierung des Problems, Sammlung der Daten, Filterung der Evidenz, Unterscheidung zwischen unterstützenden und entgegengestellten Argumenten und schlussendlich die Entscheidung. Mit dem Schwerpunkt der Evidenz-Filterung befassen wir uns mit dem Feld der Argument-Extraktion auf Satz-Ebene. Gegeben sei ein Thema und ein Satz, mit dem Ziel, den Satz hinsichtlich seiner arugmentativen Relevanz des Themas betreffend zu klassifizieren. Zumal Argumente externes Wis-

[1] Some of our results currently are under peer-review as part of a paper.
[2] Our code is accessible at `https://github.com/PatrickAbels8/GraphArgumentMining`.

sen benötigen, um gefunden und verstanden zu werden, präsentieren wir in dieser Arbeit ein graphbasiertes Modell, das Wissen aus strukturierten und unstrukturierten Quellen extrahiert. Da insbesondere unstrukturierte Daten für gewöhnlich kontextunspezifische Rohdaten sind, nutzen bisherige Arbeiten entweder nur kleine Mengen an Daten oder nur einen Typ von Daten. Wir nutzen die Latent Dirichlet Allocation, um satz- und themenspezifisches Wissen aus der Datenbank Wikidata zu extrahieren. Mittels Wort- und Satzvektoren des GloVe-Modells bewerten wir gefundene Entitäten von Wikidata hinsichtlich ihrer Relevanz zum gegeben Satz durch Berechnung der Kosinus-Ähnlichkeit und bauen so iterativ einen Wissensgraphen auf. Um die Unvollständigkeit strukturierter Wissensbasen aufzugreifen, konstruieren wir einen zweiten Graphen basierend auf den auf Google gefundenen Artikeln zum gegebenen Thema mittels des OpenIE-Modells. Indem wir beide Graphen verschmelzen, erhalten wir ein Graph-basiertes Modell, welches, wie unsere Ergebnisse zeigen, erfolgreich von strukturierten und unstrukturierten Daten profitiert. Auf dem UKP Sentential Argument Mining Corpus erreichen wir 84.16% Genauigkeit und 65.60% F1.

Acknowledgement

First and foremost I wish to thank my family for always supporting me. I am especially thankful for Dr. Zahra Ahmadi, Dr. Sophie Burkhardt and Benjamin Schiller for all the conversations, feedback, advices and suggestions in each stage of this thesis. Also, I am extremely grateful for the support of my colleagues Steffen Eiden, Cedric Luger, and Lukas Moos, only to name a few.

Contents

1 Introduction

The emerging field of argument mining aims to gather data with an argumentative context regarding a topic of interest, and by this, support decision making. Following Stab *et al.* [35], we define an argument as a combination of a topic and a sentence holding evidence towards this topic (Figure 1.1). However, without accessing some relevant world knowledge, it is difficult to understand the arguments. We tackle this problem by building a local knowledge graph from structured data (Wikidata) and unstructured data (Google search) for each sentence and extracting paths leading from one token to another. We use latent Dirichlet allocation (LDA) [3] to improve the quality of connections in the knowledge graph by comparing the context of a corpus, regarding the given sentence and topic, with the properties that each Wikidata entity may be connected with.

1.1 Background

Early work introduced world knowledge into NLP-tasks by harnessing manually constructed knowledge bases like Wikidata[1] or DBPedia[2] [4, 13]. Soon after realizing the incompleteness of structured knowledge sources, authors provided world knowledge from unstructured data such as Wikipedia[3] [32]. Having problems with noise in unstructured sources, more recent work combines both structured and unstructured data [7, 23, 27]. Motivated by the similarity of argument-based tasks, utilizing and enhancing pre-trained language models like BERT [10] has gained huge popularity [38, 39, 40, 31, 14, 16, 20, 21]. Our work aims to establish a proof of concept for knowledge graph aided argument mining.

1.2 Motivation and Problem Statement

In this thesis we focus on the problem of making a decision. Decision-making can be separated into five steps: First, we have to identify the problem. This can be the choice of wearing school uniforms, legalizing marijuana or aborting a pregnancy.

[1] https://www.wikidata.org/
[2] https://www.wiki.dbpedia.org/
[3] https://www.wikipedia.org/

Topic: *Cloning*
Sentence:
DNA cloning has been used in genetic engineering to create plants that offer better nutritional value.

$DNA \xrightarrow{WIKIFIERED} DNA\ (Q7430) \xrightarrow[P361]{part\ of}$
$DNA\ 3'\ dephosphorylation\ (Q22278441) \xrightarrow[P31]{instance\ of}$
$Entity\ obsoleted\ in\ Gene\ Ontology\ (Q93740491) \xrightarrow{MATCH} Gene\ therapy$
$\xrightarrow{MATCH} genetic\ engineering$

Fig. 1.1: A sample topic and supporting argument as input (top) and a path connecting concepts within the sentence as output (bottom).

Second, we need to collect data regarding this specific problem from different knowledge sources. Third, in order to solve the problem we need to extract evidence from the data, fourth, identifying pro and con arguments. These arguments help us assessing the benefits and drawbacks of, fifth, making the final decision. Specifically, we focus on step three being the extraction of evidence from given sentences to later help labeling them as arguments supporting or opposing a topic. Using the UKP Sentential Argument Mining Corps [34] we have access to instances containing a topic and a sentence each. Our goal is to build a model that takes a topic and a sentence as input and returns evidence regarding this specific input. Figure 1.1 shows a sample argument supporting the topic of cloning. Following statements like "DNA is a part of DNA 3' dephosphorylation" on Wikidata, we find a path that connects two concepts found in the sentence, representing evidence of the given sentence: an *evidence path*.

Our contributions of this thesis can be summarized as follows:

1. We develop a graph-based approach leveraging evidence from structured knowledge bases via latent Dirichlet allocation.

2. We propose a dynamic breadth-first search algorithm using word embeddings to create a sparse knowledge graph.

3. We introduce a method to enrich a knowledge graph with unstructured data from Google via OpenIE.

4. We achieve an average accuracy of 84.16% and F1-score of 65.60% on the UKP Sentential Argument Mining Corpus.

The thesis is structured in the following way:

Chapter 2

We will present the current state of the art in this chapter. The approaches will be organized in three categories: (1) using only structured or unstructured data, (2) using both simultaneously and (3) enhancing pre-training models with external knowledge.

Chapter 3

In this chapter the conceptual background is discussed. It starts with basic definitions like TF-IDF and other textual measures along with models and algorithms like LDA and OpenIE used in the methods presented in this thesis. The chapter goes over to knowledge graphs, introducing the knowledge base Wikidata and the word representation model GloVe. It ends with a brief look into the field of argument mining.

Chapter 4

This chapter presents the investigated strategies for property selection via topic modeling, efficient structured knowledge retrieval via word embeddings and enrichment of the graph with unstructured knowledge via OpenIE.

Chapter 5

Experimental results of several combinations of the presented methods on the UKP Sentential Argument Mining Corpus [35] datasets will be discussed.

Chapter 6

The thesis finishes with further considerations to improve on the presented methods in future work.

2 Related Work

Approaches on solving argument mining problems using external knowledge can be separated into different groups. First, there are approaches that only use one type of data: either structured data or unstructured data. Structured data is machine- or human-generated data in a knowledge base that is efficiently searchable via queries. Unstructured data is basically everything else, mostly raw textual data. Second, authors try using both structured and unstructured data at the same time. Last, some approaches enhance pre-training models with external knowledge.

2.1 Structured or unstructured data

In the early stages of argument mining it quickly became clear that any task, whether argument classification or question answering, is easier with the necessary world knowledge to understand arguments [32]. Over the last years we have witnessed a variety of approaches tackling this problem by extending models with external knowledge.

Botschen *et al.* [4] use two kinds of knowledge: event knowledge about prototypical situations from FrameNet[1] and factual knowledge about concrete entities from Wikidata, a collaboratively constructed knowledge base encoding world knowledge in triples. In Figure 1.1 for example, the world knowledge "DNA is a part of DNA 3' dephosphorylation" is encoded to the triple ("DNA", "part of", "DNA 3' dephosphorylation"). FrameNet is a lexical-semantic resource that normalizes the given sentence by assigning related tokens like "companies" and "corporations" to the same frame. Wikidata maps them to entities with detailed information in form of semantic and ontological connections. Figure 2.1[2] demonstrates how their framework works on sample inputs. However, they point out that world knowledge alone might not be enough to improve on argumentative reasoning and that logical analysis was needed. Also, they strongly focus on the given sentence regardless of the general topic it is referred to.

[1] https://framenet.icsi.berkeley.edu/fndrupal/
[2] https://www.aclweb.org/anthology/W18-5211.pdf

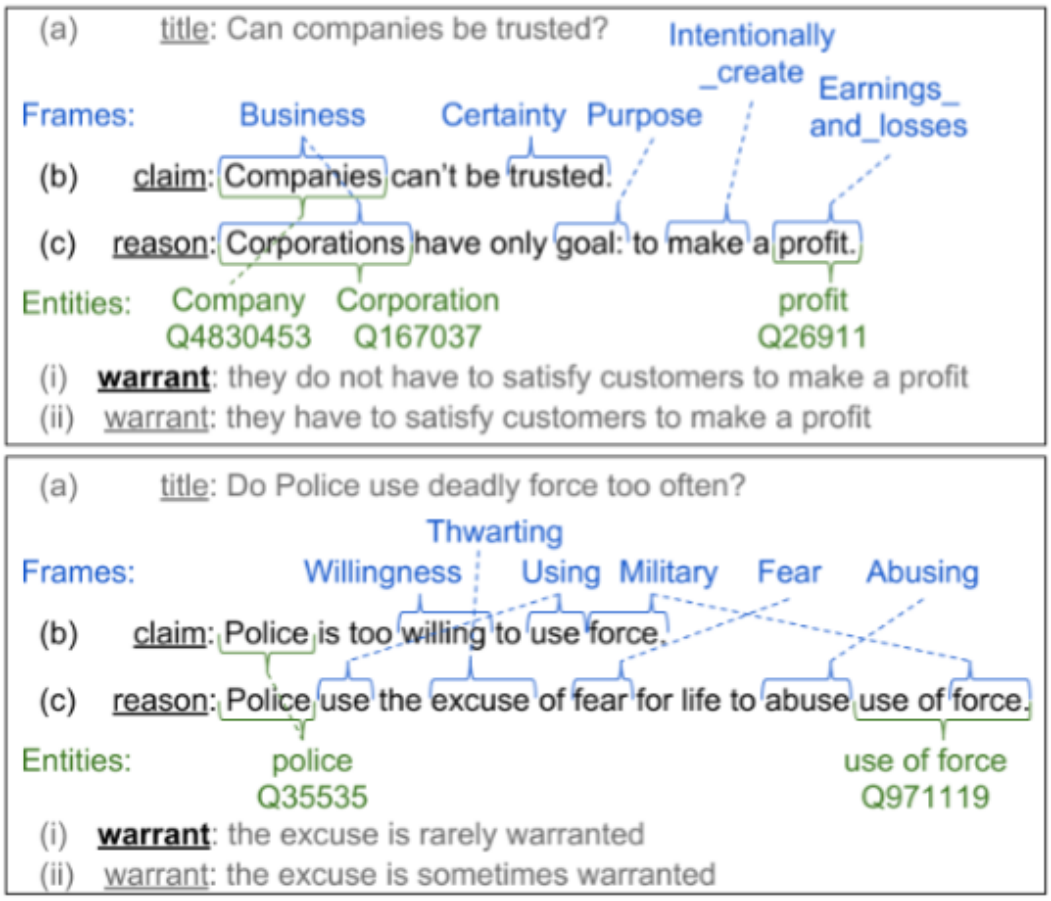


Fig. 2.1: Testing instances from the ARC dataset containing a claim and a reason each are annotated with Wikidata entities and FrameNet frames.

Fromm *et al.* [13] instead classify potential arguments by also integrating additional knowledge from DBPedia about the topic, highlighting the importance of the context between argument and topic. Furthermore, by integrating contexts from word embeddings, knowledge graph embeddings and models pre-trained on other tasks they show that considering topics leads to better understanding of the context of potential argument.

We improve on these ideas [4, 13] by (1) developing techniques to keep the context of world knowledge and the given sentence closely related, (2) analyzing additional information via natural language metrics like the tf-idf and (3) focusing on the sentence as well as the topic to the same extent.

Other than the previous approaches, Potash *et al.* [32] use Wikipedia articles to provide external knowledge via unstructured data. They compare three types of Wikipedia corpora: 30 hand-picked topic-specific articles, roughly 38k randomly chosen articles and the combination of both. All types of corpora appear to provide meaningful additional knowledge, although, none of the three performs exception-

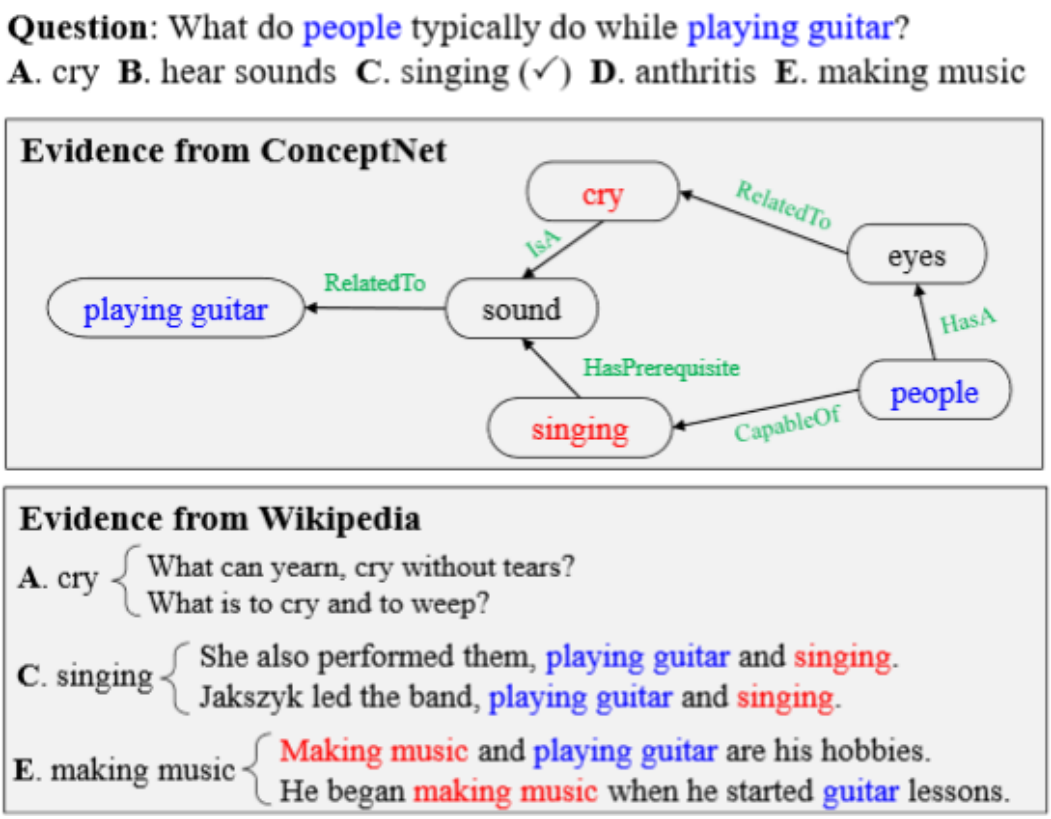


Fig. 2.2: An example from the CommonsenseQA dataset. ConceptNet evidence helps pick up choices (A, C) and Wikipedia evidence helps pick up choices (C, E). Combining both evidence derives the right answer C. Words in blue are the concepts in the question.

ally well. Especially naive models have trouble dealing with the noise in large corpora.

We capitalize on these findings by extracting unstructured data from manually picked articles based on PageRank [28].

2.2 Structured and unstructured data

More recent work that combines both structured and unstructured data proposes new benchmarks on various tasks, including question answering [7, 23], and document classification [27].

Clark *et al.* [7] solve non-diagram multiple choice Science exams with over 90% (8th grade) and 83% (12th grade) accuracy. Their Aristo project consists of three different methods: (1) statistical and information retrieval methods, including searching the exact question in structured datasets, (2) reasoning methods using semi-structured

data via OpenIE [24], and (3) large-scale language model methods such as ELMo [30] and BERT [10].

In a similar graph-based approach but in a different problem, Lv *et al.* [23] build two graphs on structured data from ConceptNet[3] and unstructured data from Wikipedia for the Commonsense Question Answering problem, achieving an accuracy of 75.3% on the CommonsenseQA dataset [36]. Their framework is visualized in Figure 2.2[4]. While they select top 10 sentences regarding the given query as Wikipedia evidence via the Elastic Search engine[5], we rely on several hundred of topic-specific sentences from Google search ranked via PageRank [28]. Furthermore, instead of ConceptNet, we use Wikidata as a source of structured data, hence, we use knowledge from Wikimedia[6]. Additionally, their rules for two nodes of the knowledge graphs being connected are relatively strict: Either one is contained in the other, or they only differ in one word. Focusing on unlabeled data, we do not want two nodes to be this restricted. Instead, we require only one common word between two nodes which adds more flexibility. While they build a graph from structured data with each statement being a node and they topology sort it to avoid cycles, we consider entities as vertices and relations as edges, having no need to sort. With their restrictive search, Lv *et al.* consider 20 nodes in structured data and 10 sentences from unstructured data, while both of our graphs from Wikidata and Google cover up to several hundred nodes.

Ostendorff *et al.* [27] enrich BERT [10] with the author information via Wikidata to improve book classification. However, they struggle to select the relevant properties to traverse the Wikidata knowledge graph. Instead, they use pre-trained graph embeddings as author representations trained on the full Wikidata graph. In contrast, we use LDA to filter the properties that match the input-specific context.

2.3 Pre-training with external knowledge

While the above approaches build upon pre-trained language models, other approaches enhance language models, more specifically BERT, with external knowledge [14, 16, 21, 31, 38, 40, 39, 20]. BERT, as the current state-of-the-art model, is pre-trained on two objectives simultaneously: masked language modeling and next sentence prediction. He *et al.* [16], for example, feed subgraphs extracted from knowledge graphs into a transformer-based model. Thereby they learn the kn-

[3] http://conceptnet.io
[4] https://arxiv.org/pdf/1909.05311.pdf
[5] https://www.elastic.co/
[6] https://www.wikimedia.org/

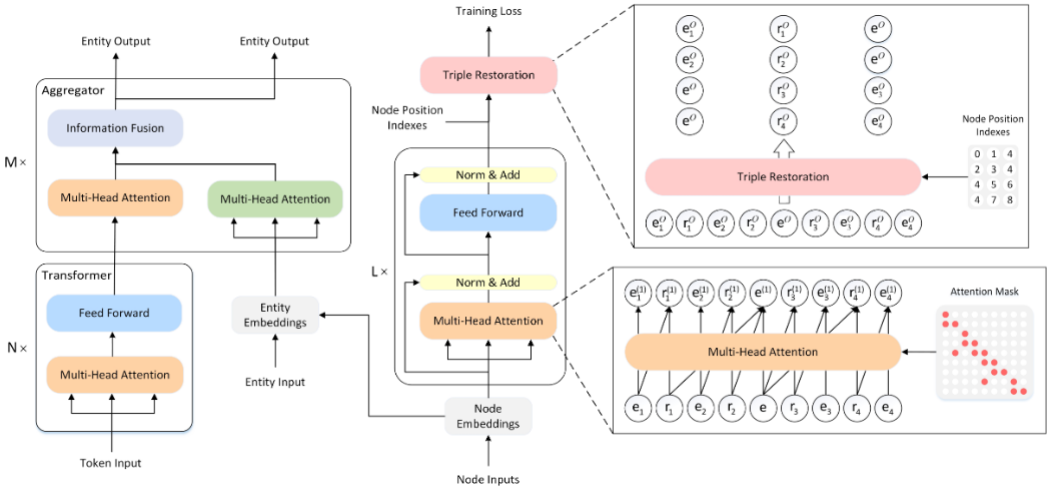


Fig. 2.3: Entity information from a knowledge graph is incorporated in a pre-trained language model (left). A KG-Transformer model (right) utilizes a training sample to describe the knowledge representation learning process.

woledge embeddings to incorporate pre-trained language models, as demonstrated in Figure 2.3[7].

[7] https://arxiv.org/pdf/1912.00147.pdf

3 Basic Concepts

In this chapter we describe the concepts, models and algorithms used in the investigated methods. Each subsection starts with a brief introduction to the field of work, and explains the specific concepts in hindsight that we use in this thesis. Additionally, we discuss some alternatives to put our methods into context. First, we introduce basic concepts in the field of text analysis. These include the word embeddings, topic modeling and text annotation. Second, we investigate knowledge graphs, specifically focusing Wikidata. Third, we take a brief look into the huge field of argument mining.

3.1 Text Analysis

The field of text analysis is huge. Since one problem focused in this thesis is finding the optimal choice of properties regarding a batch of concepts within a sentence, we are interested in several similarity measuring concepts. In terms of measuring the importance of a word to a document in a batch of documents we explain the TF-IDF. To measure the cohesion within different concepts we explain the cosine similarity. Finally, we demonstrate latent Dirichlet allocation as a topic model and OpenIE as a framework to retrieve knowledge from unstructured data.

3.1.1 TF-IDF

A measure often used in text mining is the so called “term frequency-inverse document frequency”, or short TF-IDF[1]. Given a list of properties and their descriptions we aim to select some of them given a specific word t. Obviously a word is more likely to appear in a longer text. Also, the word might appear in every property description and might therefore be meaningless. We want to address these two points by measuring the importance of a word to a specific property by looking at the list of descriptions as a corpus. While there are several measurements to choose from, the TF-IDF combines both issues by valuing the number of appearances in the document higher but the frequency in the whole corpus lower.

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

Formally, given a corpus $C = \{D_1, ..., D_n\}$ containing n documents, the "normalized term frequency" is defined as

$$TF(t, D) = \frac{\text{number of times } t \text{ appears in } D}{\text{number of words in } D}$$

and the "inverse document frequency" as

$$IDF(t, C) = \ln \frac{n}{\text{number of documents containing } t}.$$

Finally, the resulting tf-idf measuring the weight of the word t in the document $D \in C$ is then calculated as

$$TF - IDF(t, D, C) = TF(t, D) \cdot IDF(t, C).$$

3.1.2 Word Embeddings

Word and sentence representation has evolved into one of the leading fields of interest in NLP, due to its enormous area of application. We aim to compare words and sentences in their contextual relation. Therefore, we use a model that represents each word with a vector. If words are semantically and syntactically similar, their vectors have a smaller distance and angle, as visualized in Figure 3.1[2]. Working with knowledge graphs covering several hundreds of nodes, we value low computational costs. Though, comparing several word vectors to one specific sentence vector, we (a) calculate a vector representing a sentence once before we build the graph to that specific sentence. Also, we (b) do not utilize a heavyweight embedding model, such as ELMo (3072 dimensions) [30]. Since global matrix factorization methods such as latent semantic analysis (LSA) [9] do poorly on word analogy tasks, and local context window methods such as the skip-gram model [25] do not utilize statistics of the corpus, we use GloVe (50 dimensions) [29]. GloVe aims to solve both shortcomings, achieving state-of-the-art performance on several tasks. For better comparison of sentence and word, we choose to not use heavyweight sentence embedding models, such as Skip-Thought (4800 dimensions) [19], InferSent (4096 dimensions) [8] or USE (512 dimensions) [6]. Instead, staying in the same vector space, we calculate the vector of a sentence by averaging its word vectors.

GloVe

To sparsen our graph, we use the Global Vectors for Word Representation: GloVe[3]. As described in [29], their model leverages statistical information by training on

[2] https://nlp.stanford.edu/projects/glove/
[3] https://nlp.stanford.edu/projects/glove/

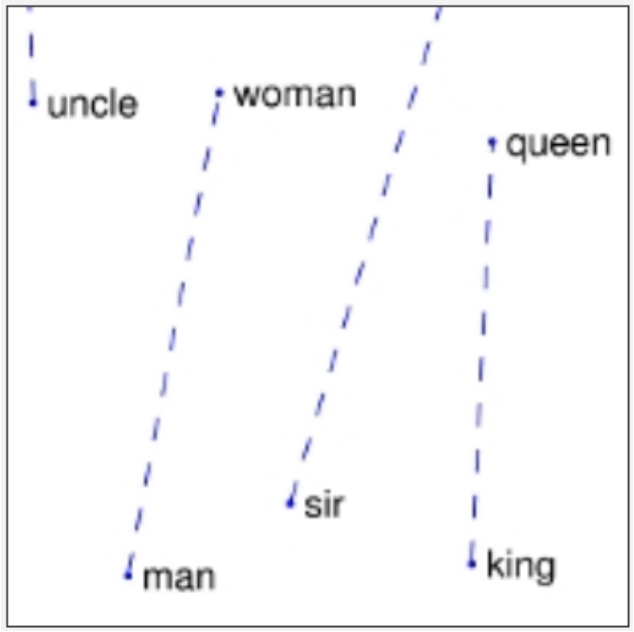


Fig. 3.1: The GloVe model maps words to vector so that the differences and angles of similar word pairs are close to equal.

aggregated global word-word co-occurrence statistics from a corpus. They perform especially well on word analogy, word similarity and named entity recognition tasks.

Sentence Embeddings

To be precise, we represent a sentence $s = \{t_1, ..., t_m\}$ containing m words with a real-valued vector

$$v_s = \frac{\sum_{i=1}^{m} v_{t_i}}{m}.$$

Cosine similarity

To measure the cohesion of the given sentence s and a word w using their representing vectors v_s and v_w the two most common approaches are judging their orientation and magnitude. Both have their benefits and drawbacks, thus we discuss why we mainly focus orientation.

Measuring orientation, the cosine similarity between two non-zero vectors $v = (v_1, ..., v_k)$ and $w = (w_1, ..., w_k)$ is defined as

$$cos(v, w) = \frac{\sum_{i=1}^{k} v_i \cdot w_i}{\sqrt{\sum_{i=1}^{k} v_i^2} \cdot \sqrt{\sum_{i=1}^{k} w_i^2}}.$$

Measuring magnitude, the euclidean distance between two vectors $v = (v_1, ..., v_k)$ and $w = (w_1, ..., w_k)$ is defined as

$$d(v, w) = \sqrt{\sum_{i=1}^{k} (v_i - w_i)^2}.$$

Instead of the distance of two vectors, their cosine is bound on the interval [-1,1] with -1 meaning exactly opposite, 1 meaning the same and 0 meaning orthogonality. Also, angles capture the similarity of concepts while disregarding the absolute magnitude of their vectors. This might become handy when working with relatively low-dimensional vectors.

3.1.3 Topic Modeling

One task of this thesis is to understand the context behind concepts found in a given sentence and a topic. Furthermore, we aim to find words, specifically Wikidata properties, that are relevant to this context. Hence, considering the Wikipedia articles of the concepts as a collection of documents, we have a text corpus to work with. Focusing automatic corpus summarization and visualization, we chose latent Dirichlet allocation [3].

Latent Dirichlet Allocation

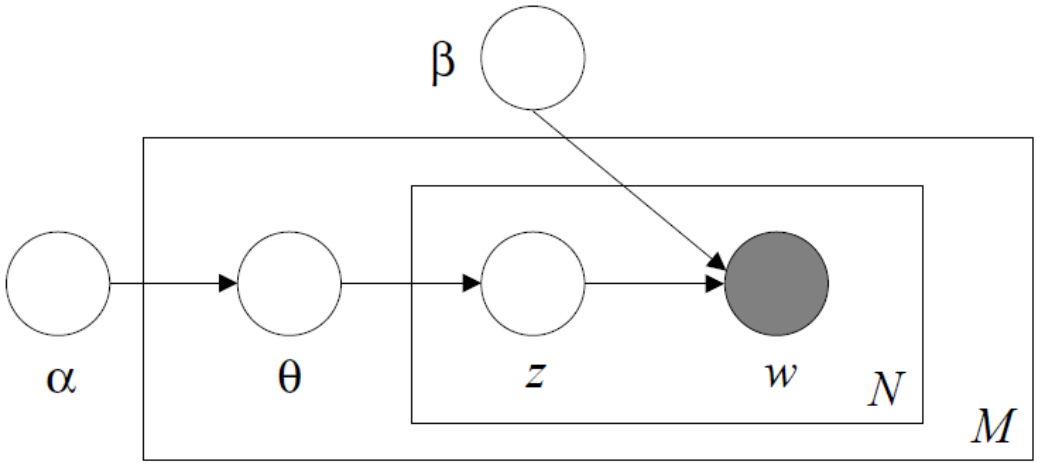


Fig. 3.2: Graphical model representation of LDA. While the outer box represents the documents in the corpus, the inner box represents the topics and words in a document.

Blei *et al.* [3] present a generative probabilistic model tackling the problem of modeling text corpora. They aim to assign high probabilities to both given documents and similar ones. Basically, they assume that (1) documents are represented as a random mixture over latent topics and (2) topics are characterized by distributions over words.

Formally, they define a $word$ denoted by w as an item from a vocabulary of size V, a $document$ denoted by $\mathbf{w} = (w_1, ..., w_N)$ as a sequence of N words, and a $corpus$ denoted by $D = \{\mathbf{w}_1, ..., \mathbf{w}_M\}$ as a collection of M documents. As shown in 3.2, the parameters needed are α, a k-vector with components $\alpha_i > 0$, and β, a $k \times V$ matrix where $\beta_{ij} = p(w^i = 1|z^i = 1)$, for a known and fixed k. Each $\mathbf{w} \in D$ is then generated by:

---

1 Choose $N \sim$ Poisson(ξ)
2 Choose $\theta \sim$ Dir(α)
3 **for** $n = 1 \rightarrow N$ **do**
4 Choose a topic $z_n \sim$ Multinomial(θ)
5 Choose a word w_n from $p(w_n|z_n, \beta)$

Following the paths of the graphical representation, a corpus containing M documents with N words generated by the corpus-level parameters α and β via the document-level variables θ_d and the word-level variables z_{dn} and w_{dn} has a probability of:

$$p(D|\alpha, \beta) = \prod_{d=1}^{M} \int p(\theta_d|\alpha) \left(\prod_{n=1}^{N_d} \sum_{z_{dn}} p(z_{dn}|\theta_d) p(w_{dn}|z_{dn}, \beta) \right) d\theta_d.$$

Using an alternating variational expectation-maximization (EM) procedure they now find α and β that maximize the log likelihood of the given corpus:

$$\ell(\alpha, \beta) = \sum_{d=1}^{M} \log p(\mathbf{w}_d|\alpha, \beta)$$

3.1.4 OpenIE

Focusing information extraction of unstructured data, Angeli *et al.* [2] present a system that produces relation triples. By applying several patterns they extract self-contained clauses from long sentences. Via natural logic inference they determine the maximally specific set of triples representing the clauses. Figure 3.3[4] demonstrates this procedure on a sample sentence.

[4] `https://nlp.stanford.edu/software/openie.html`

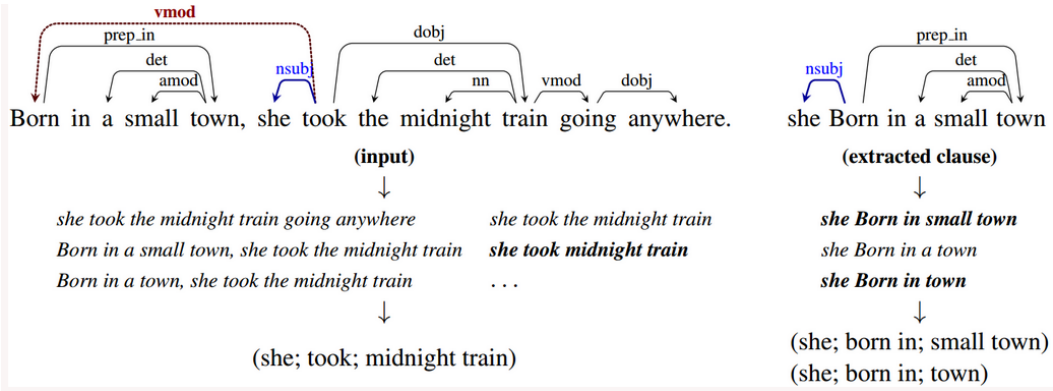


Fig. 3.3: Illustration of OpenIE annotating an example sentence with a set of triples.

3.2 Knowledge Graphs

When working with structured and unstructured data, separating useful from useless knowledge is not only hard for humans, but also for machines. Therefore, it is crucial to model the complex data while efficiently storing it. Knowledge graphs, as a set of vertices connected by a set of edges, are best at accomplishing both tasks. Wikidata, Freebase, and DBpedia, only to name a few, are publicly available data sources containing huge amounts of real-world knowledge. Figure 3.4[5] shows how querying these knowledge sources can help answering questions regarding multiple cultural topics.

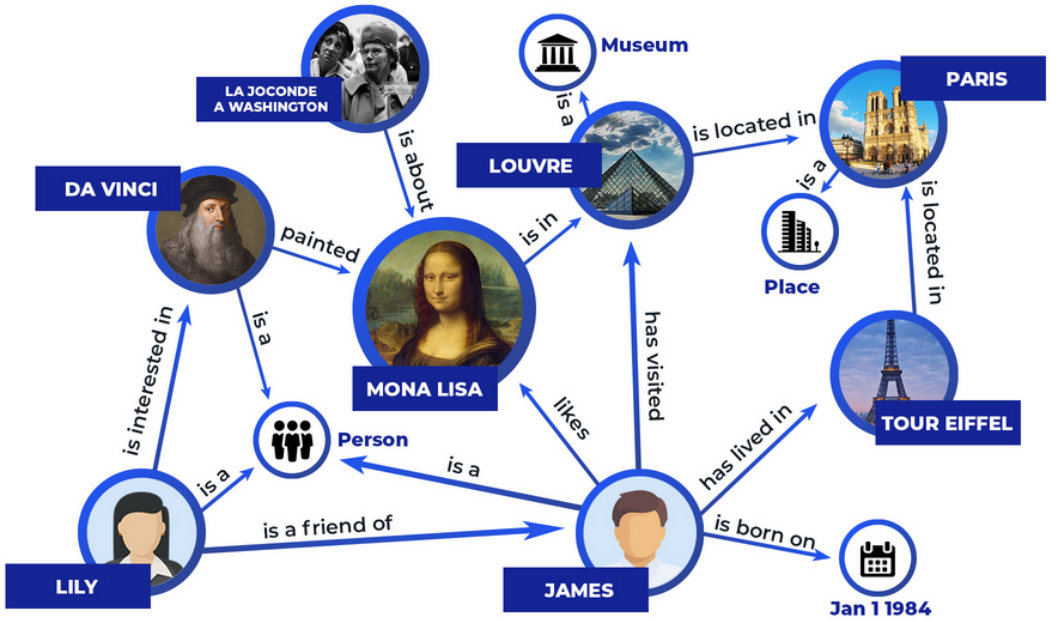


Fig. 3.4: Entities covering several topics, like culture, art and history, and their relations can be visualized as a knowledge graph.

[5] https://yashuseth.blog/2019/10/08/introduction-question-answering-knowledge-graphs-kgqa/

3.2.1 Wikidata

Our first main source of knowledge is the collaboratively constructed knowledge base Wikidata[6]. We chose Wikidata because it is free, collaborative, multilingual, and its broad community curation ensures a high data quality [12]. Wikidata currently contains more than 88 million items[7] and 5519 properties[8]. Beyond a label (e.g. "Douglas Adams"), identifier (e.g. "Q42"), description (e.g. "english writer and humorist"), and aliases (e.g. "Douglas Noel Adams, Douglas Noël Adams, and Douglas N. Adams"), each item has a set of statements linked to other items. A statement, expressing a semantic or ontological connection, can be described as a binary relation between entities (e.g. P69(Q42,Q691283) representing the fact that Douglas Adams (Q42) got educated at (P69) St John's College (Q691283)). Formally, we describe Wikidata as a graph $G := (E, R, S)$ with $S = \{r(e_1, e_2) | r \in R, e_1, e_1 \in S\}$. One way to obtain knowledge from Wikidata is by querying data through their SPARQL[9] endpoint. By submitting POST requests with queries of the form in Figure 4.3 we select every distinct (entity-property-entity)-instance (line 3) where the first entity (subject of the statement) is "Q42" ("Douglas Adams") and the property (relation of the statement) is "P69" ("educated at") (line 4), which leads to all the entities Douglas Adams got educated in. We also gain access to the label of these entities (line 5) in the english language (line 6). The first two lines are necessary to access the entities and properties by their identifier.

3.3 Argument Mining

Making a decision as a human can be hard. When faced with crucial decision, we first try to understand the matter of concern. We ask friends or experts, read articles and collect as much data as possible regarding the given context. To differentiate between the knowledge being important to this specific decision and useless knowledge, we extract the evidence from the collected data and identify whether the arguments support or oppose an optional decision. Finally, we make our decision. In the field of machine learning, we aim to teach machines making their decision. Considering these human steps, the most error-prone and time-consuming steps are data collection and extracting the evidence from the given data. As a subfield of natural-language processing, Argument mining aims to automatically extract and identify argumentative structures from textual data. We lean onto the definition by Stab *et al.* [35] who define an *argument* as the combination of a topic and a sentence holding evidence towards this topic.

[6] https://www.wikidata.org/
[7] https://www.wikidata.org/wiki/Wikidata:Statistics
[8] https://www.wikidata.org/wiki/Wikidata:Database_reports/List_of_properties/all
[9] https://query.wikidata.org/

In the early stages of argument mining, arguments were mainly focused on a discourse level. Emphasizing the structural essence behind single sentences, these approaches did not consider topic-coherence [1, 11]. Later approaches identify whether a sentence supports or opposes a specific topic but do not mind if the sentences are not even evidence holding arguments [26] or only cover very specific types of sentences not showing any robustness [37, 22, 33]. Combining all these ideas, Stab *et al.* [35] focus cross-domain argument mining by investigating heterogeneous texts covering different topics.

4 Methods

We now present our framework in four steps: (1) Our baseline provided by Benjamin Schiller[1], (2) LDA-driven property selection, (3) word embeddings for the graph built on structured data by Wikidata, and (4) building a second graph based on unstructured data from Google searched articles. The whole process is visualized in Figure 4.1, the topic and sentence are mapped to the Wikidata entities via the Wikifier [5], an online entity linker, and the local knowledge graph built from Wikidata. Wikidata, an open structured knowledge graph, stores information in the form of tuples as described earlier in Section 3.2.1. This allows us to iteratively build a local graph with Breadth-First Search (BFS) and finally extract evidence paths. We focus on two kinds of information: (1) paths between the topic and the sentence signaling topic relevance and (2) paths between tokens within a sentence to gather additional world knowledge ("evidence") for it, hence called "evidence paths" in the following.

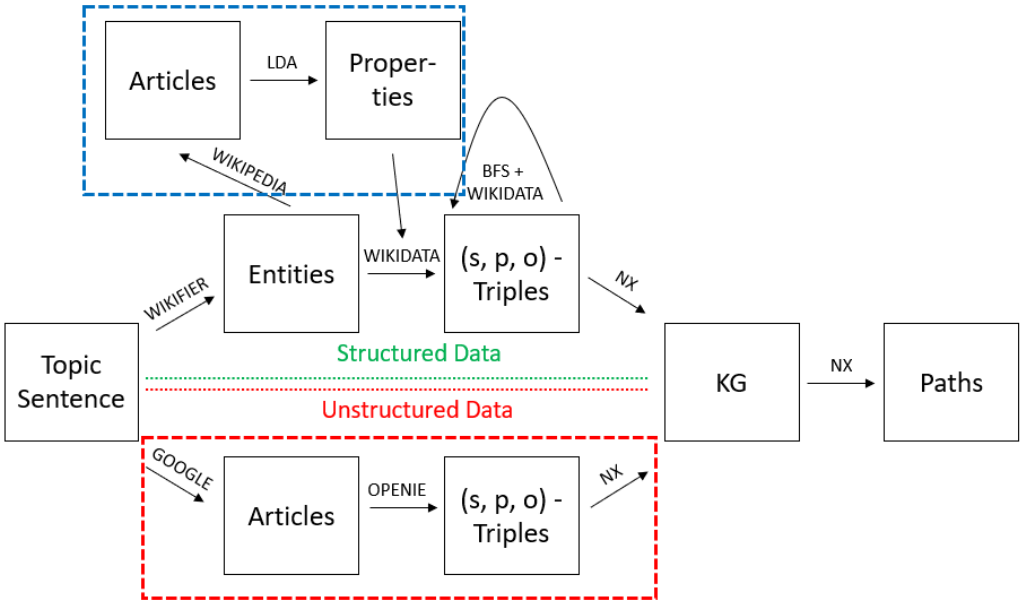


Fig. 4.1: The total framework maps a topic and a sentence to a set of paths. The baseline is supplemented by LDA-driven property selection (blue box) and unstructured knowledge enrichment via Google search (red box).

[1]Benjamin Schiller M.Sc., Doctoral researcher at Ubiquitous Knowledge Processing (UKP) Lab at the Technische Universität Darmstadt.

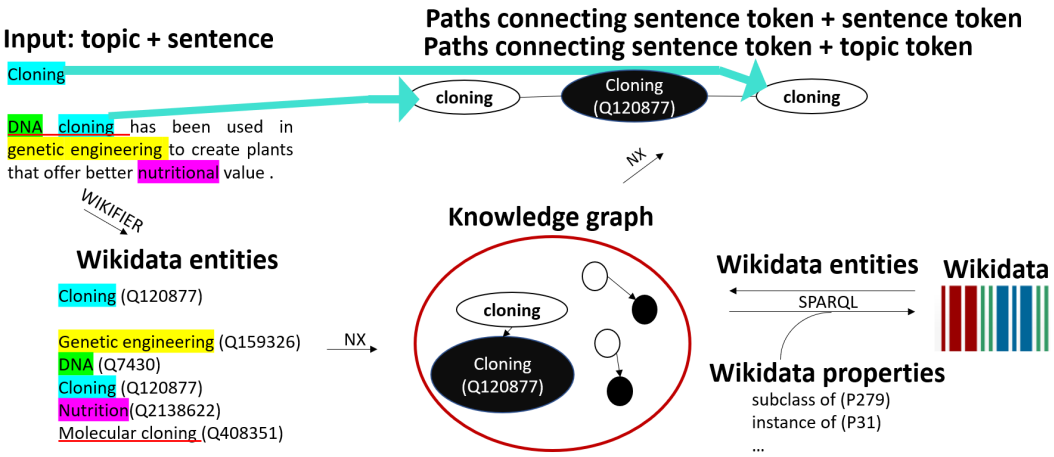
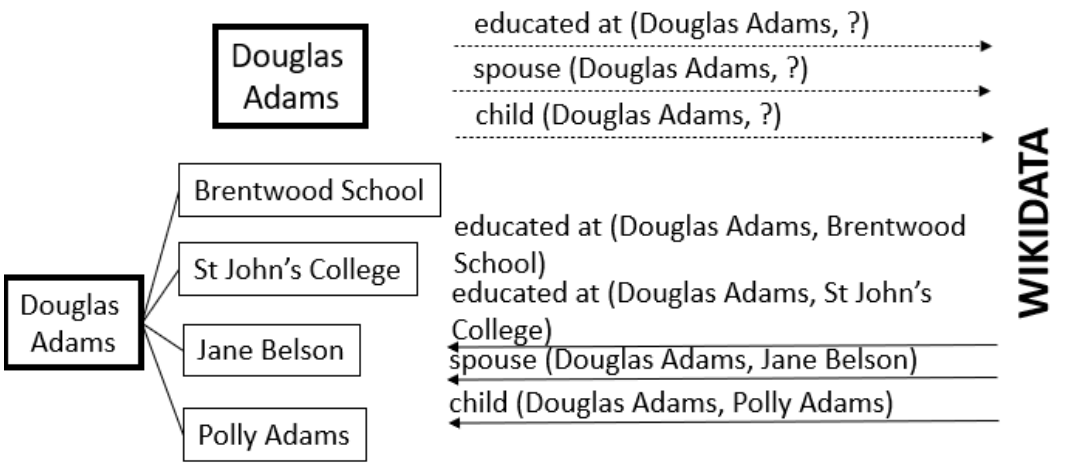


Fig. 4.2: Our main core executed on a sample input. The output is one evidence path connecting a topic-concept and a sentence-concept.

4.1 Main Core

An argument is a combination of a topic and a sentence holding evidence towards this topic which are given as input to the method. As shown in Figure 4.2, these are the inputs to the Wikifier which annotates the input document with relevant Wikipedia concepts via a PageRank-based method. The output is a JSON document containing a list of annotated Wikipedia concepts along with their corresponding Wikidata entities. As an input, we consider the top k_s concepts found in the sentence and the top k_t concepts found in the topic, each concept representing a specific token. The $k_s + k_t$ concept nodes and their corresponding $k_s + k_t$ entity nodes build the foundation of our knowledge graph. In every of the n_d iterations, we query Wikidata via SPARQL-queries for a list of entities. We desire every entity that is connected to one of the unseen entity nodes in the graph via one of the properties of interest. Figure 4.3 represents a sample Wikidata query with its returned list of object-entities as part of a node's expansion. Repeating this BFS-like expansion in every iteration, a knowledge graph representing the Wikidata-based concepts and their relations regarding the given topic and sentence is extracted. To cover both, sentence-sensibility and sentence-topic-coherence, we extract two kinds of paths from the graph using NetworkX[2]: (1) the shortest path connecting one topic-concept with one sentence-concept and (2) the shortest path connecting two sentence-concepts. These paths serve as evidence to help classify whether the given sentence is an argument to the given topic.

[2] `https://networkx.github.io/`




```
PREFIX wd:<http://www.wikidata.org/entity/>
PREFIX wdt:<http://www.wikidata.org/prop/direct/>
SELECT DISTINCT *
WHERE { wd:Q42 wdt:P69 ?o .
        ?o rdfs:label ?label .
        FILTER (lang(?label) = 'en') .
      }
```

```
Q4961791: Brentwood School
Q691283: St John's College
```

Fig. 4.3: Expansion of the node "Douglas Adams" via a SPARQL query (dashed arrow) per entity per property (top), one such query (middle), and its output (bottom).

4.2 Property Selection with Latent Dirichlet Allocation

4.2.1 Problem Statement

Building a knowledge graph over all the existing properties per node is infeasible as: (1) In the worst case, at depth n_d the graph already covers

$$\#nodes = \sum_{d=0}^{n_d} (k_s + k_t) n_p^d$$

nodes and

$$\#edges = \sum_{d=1}^{n_d} (k_s + k_t) n_p^d$$

edges with $k_s + k_t$ being the number of entity nodes that we started with and n_p the number of properties. With the BFS runtime of a graph being $\mathcal{O}(\#edges + \#nodes)$,

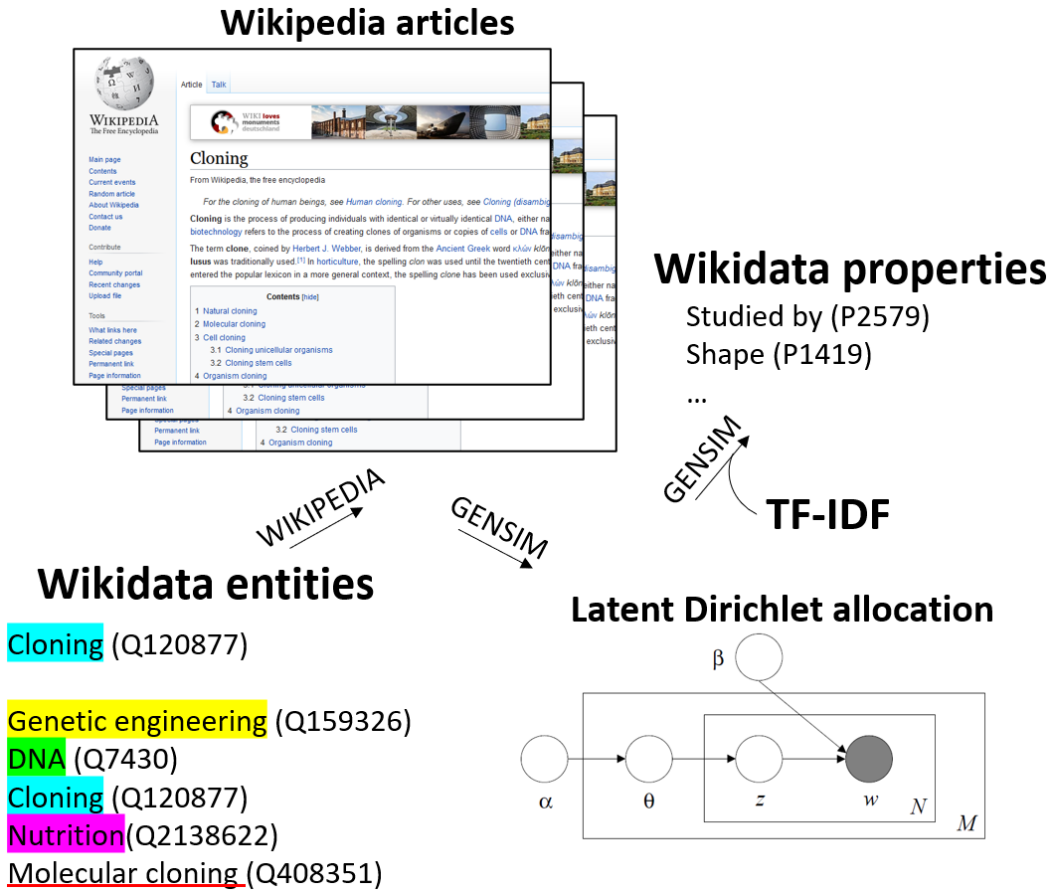


Fig. 4.4: LDA driven property selection executed on a sample input. A set of entities are mapped to a set of properties, matching the given context.

in real-world applications the search is not feasible for $n_p = 5519$[3]. (2) Most of the properties do not help gaining relevant information on the given concepts, they instead drastically increase the noise. (3) Building the graph based on a small fixed set of properties leaves out most of the available information. (4) Alternatively, taking a predetermined set of properties (e.g., the 50 most frequent ones) lacks coherence to the given context. Considering these challenges, we select relevant properties to the context of a specific sentence and topic dynamically by latent Dirichlet allocation. As shown in Figure 4.4, we train a model on the Wikipedia articles of the given concepts in the topic and the sentence via the Python library Gensim[4] and evaluate the available Wikidata properties regarding the LDA output via TF-IDF.

[3] https://www.wikidata.org/wiki/Wikidata:Database_reports/List_of_properties/all
[4] https://pypi.org/project/gensim/

Top 5 topics from LDA on the corpus [Romeo, Juliet, Romance]

...	...	Juliet, Romeo, the, age, love, year, in, wall, woman, English, portrayed, romantic, young, Shakespeare, married, letter, story, film, verona, adaptation	...	...

ID	Label	Aliases	Description	Count
		...		
P26	spouse	Partner, husband, wife, married to, consort, marry, marriage partner, married, wedded to, wed, life partner	The subject has the object as their spouse.	97474
		...		

Fig. 4.5: We compare the output of the LDA (top) and the property descriptions of Wikidata (bottom) to select the set of properties matching the given context.

4.2.2 Data and Preprocessing

As shown in Figure 4.1, the blue box covers three steps: (1) retrieving Wikipedia articles on the given entities, (2) training an LDA model on the articles to discover properties related to this specific context, and (3) embedding the properties into the main core. While the first and third steps are lightweight in terms of preprocessing effort, in the second step, we need to find a way to compare the properties of Wikidata to the LDA output. Fortunately, Wikidata provides a list of all properties with a description. This information serves as an interface between the properties and the words representing the topics given by LDA (listed as input in Algorithm 1 under "property_descriptions"). As shown in Figure 4.5, the LDA trained on the Wikipedia articles of "Romeo", "Juliet", and "Romance" discovered five hidden topics, one of them representing the love story between the two characters. This topic is modeled as a list of words, one of them being the word "married". Wikidata's list of properties and their descriptions covers, among others, the property "spouse" which has the aliases "married to" and "married". We take both appearances into consideration when calculating the TF-IDF measured relevance of the property "spouse" in this context.

4.2.3 Algorithm

We derive the list of properties given a list of entities in four steps (Algorithm 1):

(1) Load the corresponding Wikipedia articles (lines 1-3) using the Wikipedia-API[5] to find the given entities' articles and extract the relevant texts.

(2) Train a LDA model (lines 4-5) to extract the most relevant topics for this specific context. Our main library for tasks including preprocessing and model building is

[5] `https://pypi.org/project/Wikipedia-API/`

ALGORITHM 1: PROPERTIES_PER_ENTITIES

Input: entities E, tfidf-threshold t_t, count-threshold t_c, num-topics n_t, num-properties n_p, property-descriptions P
Output: properties $\tilde{P}$

```
1  D ← ∅
2  forall e ∈ E do
3      D ← D∪Wikipedia article of e
4  L ← LDA(D, n_t)
5  T ← top-topics(L)
6  W ← ∅
7  forall t ∈ T do
8      forall w ∈ t do
9          if w ∉ W and w ∉ stopwords then
10             W ← W ∪ w
11 F ← RANK_BY_TFIDF(W, t_t)
12 P̃ ← ∅
13 forall w ∈ F do
14     forall p ∈ P do
15         if p ∉ P̃ and w ∈ p.info and p.count > t_c then
16             P̃ ← P̃ ∪ p
17 return SELECT_FREQUENT(P̃, n_p)
```

Gensim[6]. Using Gensim we remove stopwords, punctuation and numeric expressions from the article documents, lowercase and tokenize[7] them as well as lemmatize[8] the tokens. We build a dictionary (list of (token,count)-tuples) and a corpus based on the preprocessed text which we apply LDA on. Then, we extract the n_t best topics represented as a list of words each.

(3) Rank the words representing the topics (lines 6-11) by their relevance to the properties measured via TF-IDF. In order to retrieve the relevant yet indispensable words, we consider the property descriptions as a batch of documents. We build a matrix M with each cell $m_{i,j}$ being the TF-IDF of the i-th word and the j-th document as described earlier in Section 3.1.1 and rank the words by their cumulative score

$$S_i = \sum_j m_{i,j}$$

[6] https://pypi.org/project/gensim/
[7] https://www.kite.com/python/docs/nltk.RegexpTokenizer
[8] https://www.nltk.org/_modules/nltk/stem/wordnet.html

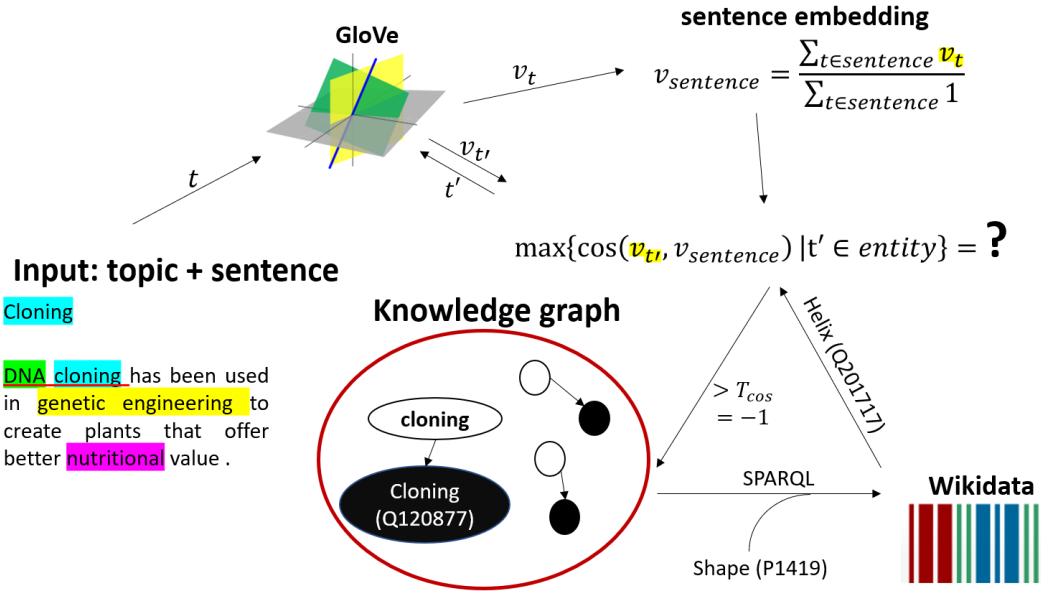


Fig. 4.6: Our dynamic BFS with word embeddings executed on a sample input creates a knowledge graph based on the similarity of new entity vectors and the sentence vector.

proceeding with only those achieving a given threshold t_t. Sticking with the example in Figure 4.5, the word "married" appears in several property descriptions, one of them being "spouse". As demonstrated in Section 3.1.1, we derive

$$\begin{aligned} S_{\text{married}} &= m_{\text{married,married name}} + m_{\text{married,spouse}} + m_{\text{married,place of marriage}} + m_{\text{married,partner}} \\ &= \frac{1}{11} \cdot \ln \frac{5519}{4} + \frac{2}{24} \cdot \ln \frac{5519}{4} + \frac{2}{29} \cdot \ln \frac{5519}{4} + \frac{2}{28} \cdot \ln \frac{5519}{4} \\ &\approx 0.6572 + 0.6025 + 0.4986 + 0.5164 \\ &= 2.2747 \end{aligned}$$

measuring the importance of "married".

(4) Extract top-related properties (lines 12-16) by searching the property descriptions for the top ranked words. We consider every property with at least one word appearing at least once. Ranked by their number of appearances, we return the n_p most frequent properties.

4.3 Knowledge Retrieval with Dynamic BFS and Word Embeddings

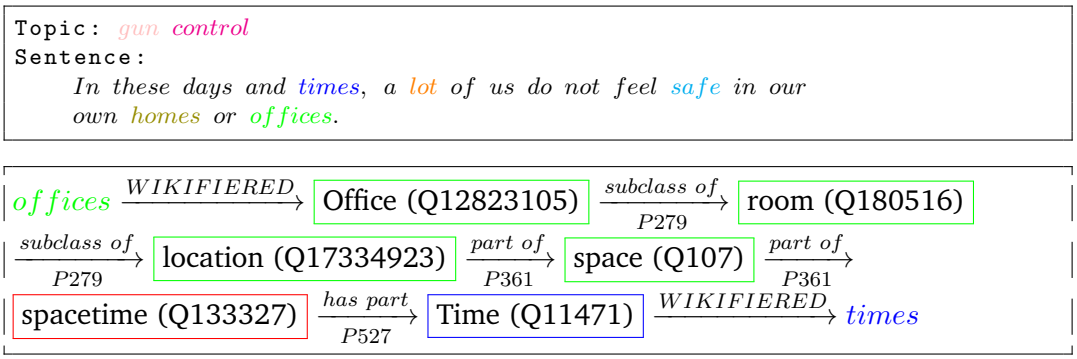


Fig. 4.7: For this sample non-argument (top), our main core with LDA driven property selection outputs a noisy path (top) connecting two concepts within the sentence.

4.3.1 Problem Statement

Only one node connection out of the context can result in a noisy path when building a graph dynamically. For instance, Figure 4.7 shows one possible path connecting the concepts *offices* and *times* within the sentence. This noisy path can only be found since the entity spacetime (Q133327) connects the offices-related concepts and the times-related concepts, even though, they do not fit into this sentence-specific context. Furthermore, if concepts are relevant with respect to more than one topic, they build a knowledge graph with a set of subgraphs, each covering the knowledge of one particular topic. Traversing such diverse graph is not optimal, instead, deeper exploration of one relevant subgraph could yield important information. As shown in Figure 4.6, we address these challenges by only expanding child nodes if the GloVe word embedding of every node (entity vector) belongs to the context of the input sentence (sentence vector) with respect to a cosine similarity threshold. In this example, the entity "Helix" consists of the one token $t' =$ helix which is mapped to the vector $v_{t'}$. Calculating the sentence vector

$$v_{sentence} = \frac{v_{\text{DNA}} + \ldots + v_{\text{value}}}{16}$$

as introduced in Section 3.1.2, we measure the context-correspondence as

$$cos(v_{sentence}, v_{t'}).$$

GloVe primarily performs well on word analogy, word similarity, and named entity recognition tasks which makes it suitable to our setting. We lower the computational complexity by not querying once per (entity, relation)-pair but per entity, enabling a theoretical speedup of the size of properties.

ALGORITHM 2: DYNAMIC_BFS

Input: sentence s, concepts C, entities E, embedding-model GV, cosine-threshold t_{cos}, max-nodes n_n, max-depth n_d
Output: graph G

1 $G \leftarrow$ MultiDiGraph()
2 **for** $i = 0 \ldots \#E - 1$ **do**
3 $\quad G \leftarrow G \cup edge(C_i, \text{“WIKIFIERED”}, E_i)$
4 $v_s \leftarrow \text{avg}(\{GV(t) | t \in \text{tokenize(lower}(s))\})$
5 $P \leftarrow$ *PROPERTIES_PER_ENTITIES*(E)
6 $d \leftarrow 0, E_{visited} \leftarrow \emptyset, E_{tovisit} \leftarrow E$
7 **while** $E_{tovisit} \neq \emptyset$ ***and*** $d < n_d$ ***and*** $\#E_{visited} < n_n$ **do**
8 $\quad d \leftarrow d + 1$
9 $\quad$ **forall** $e \in E_{tovisit}$ **do**
10 $\quad\quad E_{tovisit} \leftarrow E_{tovisit} \setminus e$
11 $\quad\quad$ **if** $e \notin E_{visited}$ **then**
12 $\quad\quad\quad E_{visited} \leftarrow E_{visited} \cup \{e\}$
13 $\quad\quad\quad$ **forall** $(e, p, e_{new}) \in$ *QUERY*(e, P) **do**
14 $\quad\quad\quad\quad$ **if** $e_{new} \notin E_{visited}$ ***and*** $e_{new} \notin E_{tovisit}$ ***and*** *COSINE_SIMILARITY*$(GV, v_s, e_{new}, t_{cos})$ **then**
15 $\quad\quad\quad\quad\quad E_{tovisit} \leftarrow E_{tovisitd} \cup \{e_{new}\}$
16 $\quad\quad\quad\quad\quad G \leftarrow G \cup edge(e, p, e_{new})$
17 **return** G

4.3.2 Data and Preprocessing

We convert the GloVe data[9] to a dictionary mapping each word to its vector. This dictionary serves as look-up model for the words in the sentence and entities, labeled in Algorithm 2 as “embedding_model”.

4.3.3 Algorithm

We map each word to its GloVe embedding and propose an augmentation method to any BFS-based graph construction in Algorithm 2. Via NetworkX[10] we initialize a directional graph and allow multiple edges between two nodes. For every concept in the given topic or sentence which is annotated by an entity via the Wikifier, we add two nodes and one edge to the graph to represent their “WIKIFIERED”-relation (line 2–3). These concept- and entity-nodes build the basis of our graph. Before further growth of the graph, we calculate the sentence vector v_s as the average of the word vectors contained in the sentence (line 4) via the GloVe model, GV. Using Algorithm 1, we receive a list of properties to build the graph (line 5). Lines 6-16

[9] https://nlp.stanford.edu/projects/glove/
[10] https://networkx.github.io/

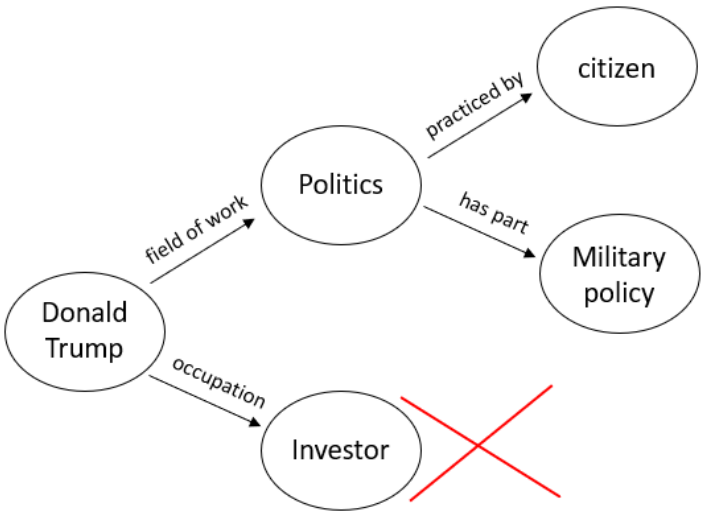


Fig. 4.8: Visualization of how word embeddings make the graph more sparse.

present a modified version of the standard BFS algorithm by querying Wikidata (line 13) and pruning the graph (line 14). To improve the query time of the increasing number of properties, we adapt the SPARQL-queries by submitting a Wikidata query per entity instead of one query per (entity, property)-pair. The returned list of (subject, predicate, object)-statements contains potential new object-entities. However, before adding a new entity e_{new} to the graph, we prove it to fulfill one of the following conditions:

$$\begin{aligned} \max\{cos(v_s, v_t) | t \in e_{new}\} &> t_{cos} \\ \max\{cos(v_s, v_t) | t \in e_{new}\} &= -1, \end{aligned}$$

where t_{cos} is a threshold and v_t are the GV-vectors of the lowercase tokens in e_{new}. We thereby assure at least one of the tokens in the new entity stays in the context of the given sentence. Being equivalent to -1 means that GV covers no token in the entity. We make sure that particular entities do not vanish. Therefore, we only exclude entities with a maximum cosine being strictly greater than -1 but less than t_{cos}. Figure 4.8 visualizes the exploration of Wikidata into the sentence-specific context for a sample input sentence "Trump uses the military to prove his manhood". Calculating the cosine similarities yields:

$$\begin{aligned} & & cos(v_s, GV(\text{“Trump”})) &> t_{cos} \\ & & cos(v_s, GV(\text{“Politics”})) &> t_{cos} \\ -1 &< cos(v_s, GV(\text{“Investor”})) &< t_{cos}. \end{aligned}$$

Therefore, the sentence-specific relevant nodes "Donald Trump" and "Politics" expand, the node "Investor" does not.

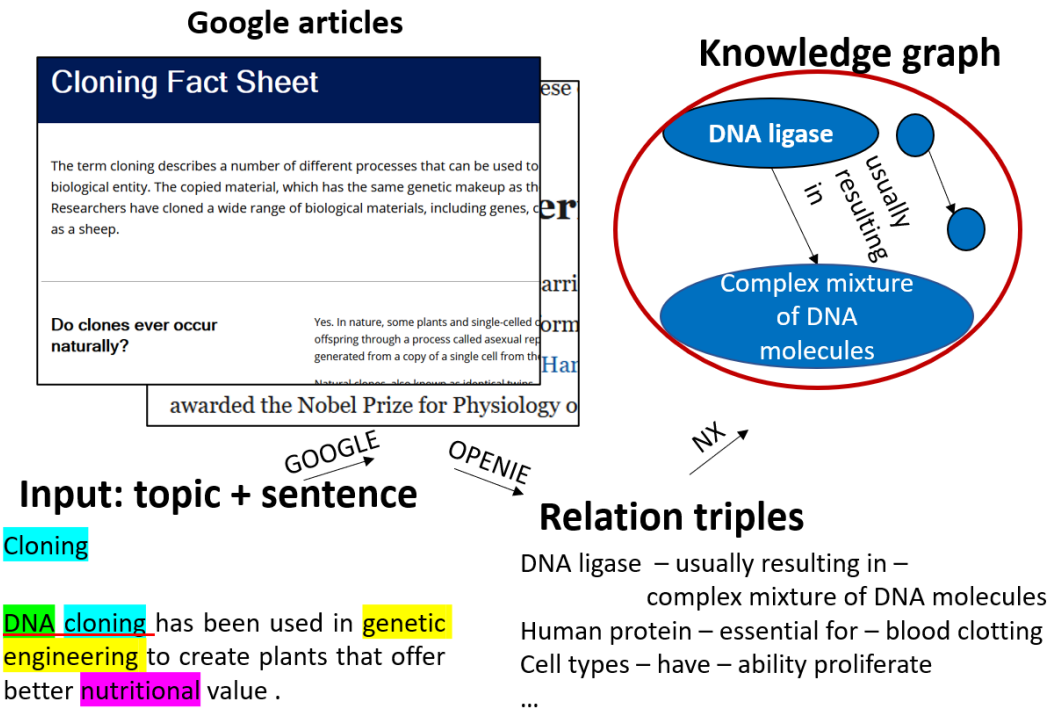


Fig. 4.9: KG of topic-specific unstructured data executed on a sample input.

4.4 Enriching the Knowledge graph with OpenIE

4.4.1 Problem Statement

Recent studies proved the importance of additional unstructured data to compensate for the incompleteness of structured knowledge-bases like Wikidata [32, 23, 7]. However, most approaches struggle with picking the relevant data and/or handling the noise [32]. Instead of using either hand-picked or randomly chosen articles [32], we consider top-ranked articles based on PageRank found in the Google searches for a given topic. Using OpenIE [24], we build a second knowledge graph and combine them gradually to handle the noise. Figure 4.9 visualizes this procedure. With the annotated triples of OpenIE on the articles on “Cloning” found on Google, useful statements like “DNA ligase usually resulting in complex mixtures of DNA molecules” are encoded to nodes and edges, building a knowledge graph.

4.4.2 Algorithm

We enrich the knowledge graph with unstructured data in 4 steps (Algorithm 3):

(1) Load the corpus. (lines 1-7) By searching Google for a given topic, we receive a list of websites ranked by Google’s PageRank algorithm (line 1). Considering the

ALGORITHM 3: ENRICH

Input: graph $G = (V_G, E_G)$, topic t, max-urls n_u, max-annotations n_a, max-chars $n_{c_{max}}$, min-chars $n_{c_{min}}$
Output: graph G

1 $U \leftarrow$ *PAGE_RANK*(t, n_u)
2 $S \leftarrow \emptyset$
3 **forall** $u \in U$ **do**
4 **forall** $s \in u.text$ **do**
5 $S \leftarrow S \cup s$
6 $S \leftarrow$ *SORT_BY_SIZE*(S)
7 corpus $\leftarrow$ *EXHAUST*$(S, n_{c_{max}}, n_{c_{min}})$
8 $A \leftarrow$ *OPENIE_ANNOTATE*(corpus)
9 **for** $i = 0 \rightarrow n_a$ **do**
10 (subject, predicate, object) $\leftarrow A_i$
11 $G \leftarrow G \cup edge$(subject, predicate, object)
12 **forall** $v \in V_G$ **do**
13 **if** *MATCH*$(v, subject)$ **then**
14 $G \leftarrow G \cup edge(v,$ "MATCH", subject$)$
15 **if** *MATCH*$(v, object)$ **then**
16 $G \leftarrow G \cup edge(v,$ "MATCH", object$)$
17 **return** G

top n_u websites, we extract each sentence with at least three words (line 5) - the minimum to let OpenIE build a triple from. For reasons like runtime, storage, and OpenIE limits we rank the sentences by their total character length (line 6).

(2) Extract (subject, predicate, object)-triples (line 8) via Stanford's information extraction framework OpenIE which takes the corpus as input. These triples representing statements are the basis of our second knowledge graph after removing duplicates.

(3) Build the graph (lines 9-11) based on the first n_a statements. If for example the sentence "In general members of politics have power" is annotated with the triple ("Members of politics", "in general have", "power"), the subject and object become nodes while the relation converts to an edge between them (line 11).

(4) Combine the two knowledge graphs (lines 12-17) $G_s = (V_s, E_s)$ and $G_u = (V_u, E_u)$ by summing up the set of edges $\mathcal{V}$:

$$\mathcal{V} = \{(u, v) \in V_s \times V_u | \exists t : t \in u, t \in v, t \notin \text{stopwords}, t \notin \text{numerics}, |e| > 2\},$$

with each node being a set of tokenized and lemmatized tokens. Figure 4.10 illustrates this procedure for a sample sentence.

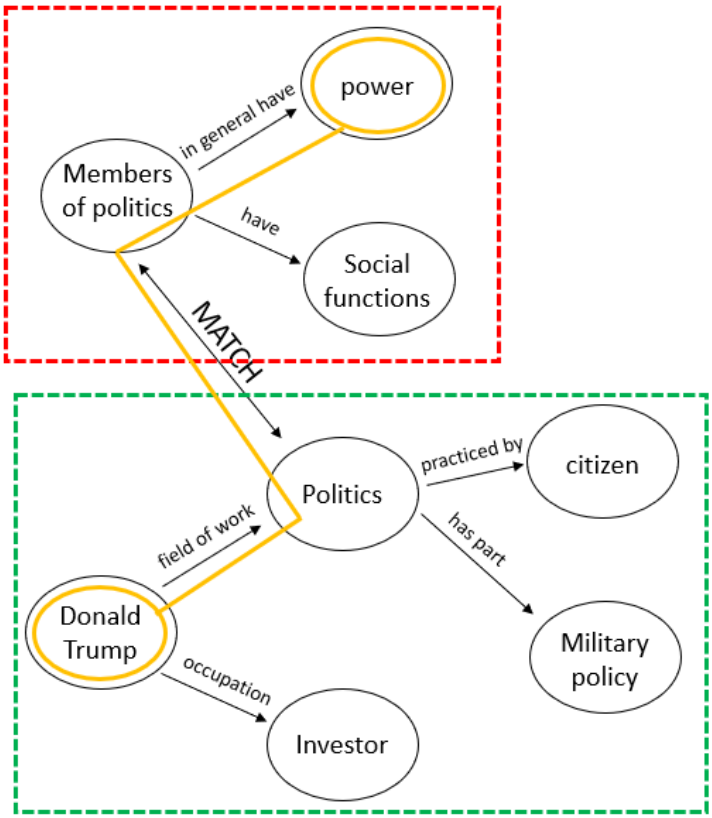


Fig. 4.10: Combination of structured data driven knowledge graph and unstructured data from Google. Given the input sentence "Donald Trump has a lot of power." the structured data will yield the green part of the graph, while the unstructured data will yield the red part of the graph. By combining the two kinds of knowledge retrieval via a "MATCH"-function, one discovers the orange path which would not have been possible with only one source of knowledge.

4.5 Conclusion

Our goal is to extract relevant paths from the knowledge graph that may help in the downstream argument mining task. The paths connect entities from the given topic and sentence. The challenge is to do this efficiently despite the huge number of entities and properties being present in the knowledge graph. Additionally, the path should be focused on a specific topic to ensure that it is a relevant path and not a connection based on random hops. We train LDA on the Wikipedia articles of each entity to choose more suitable properties for growing the knowledge graph. Using TF-IDF, we filter the highest scoring properties for the topics given by the model. Avoiding areas in the graph that are not within the given sentence context, we map each token of an entity to a specific word vector and check whether its cosine similarity with the sentence vector passes a certain threshold. Additionally, we build a second knowledge graph based on unstructured data from Google search for a given topic. Annotating the documents via OpenIE, we combine the two graphs to tackle the incompleteness of Wikidata.

5 Experimental Evaluation

This chapter is divided into three parts: First, we introduce the baseline we compare to in the task of argument identification. Second, we present the experimental settings along with the frameworks that we test. Last, we discuss qualitative and quantitative results as well as an error analysis and a case study.

5.1 Argument Identification

We model the task of argument identification as a binary classification problem. Since we explicitly do not focus on the stance identification of arguments, we do not differentiate between pros and cons. Hence, we only consider two labels, argument or no-argument. We use the following models for the prediction task:

Baseline uses a the BiLSTM from [18] that only receives the sentence as an input.

Baseline+X incorporates the collected evidence paths for each sentence using two BiLSTMs and an attention mechanism. One BiLSTM is used to flatten each path to a single vector, and the other BiLSTM encodes the sentence. Attention is used to generate a single vector from an embedded token of a sentence and all path vectors connected to that.

We define a path as a sequence $e_0, p_0, e_1, p_1, ..., e_n$, consisting of entities e and predicates p, embedded with pre-trained knowledge graph embeddings. We use a BiLSTM (shared among all paths) to reduce each path to a single vector q (last hidden state of that BiLSTM). All paths $(q_1, q_2, ..., q_m)$ for an embedded token v of an input sentence are then used in an attention mechanism to gain a final vector u. The attention mechanism is defined as follows [17]:

$$\boldsymbol{m}_i = \tanh(\boldsymbol{W}_q \boldsymbol{q}_i + \boldsymbol{W}_v \boldsymbol{v}),$$
$$f_{attention}(\boldsymbol{q}_i, \boldsymbol{v}) = \frac{\exp(\boldsymbol{w}_m^T \boldsymbol{m}_i)}{\sum_i \exp(\boldsymbol{w}_m^T \boldsymbol{m}_i)},$$

topic	sentences	no-argument	argument
abortion	3,929	2,427	1,502
cloning	3,039	1,494	1,545
death penalty	3,651	2,083	1,568
gun control	3,341	1,889	1,452
marijuana legalization	2,475	1,262	1,213
minimum wage	2,473	1,346	1,127
nuclear energy	3,576	2,118	1,458
school uniforms	3,008	1,734	1,274
total	25,492	14,353	11,139

topic	sentence	label
death penalty	We do not trust the government to deliver mail or launch a healthcare website, so why would we trust them with life and death decisions?	argument
school uniforms	But I think this, I think that local communities and states should make the decision and I feel very strongly about that.	no-argument
nuclear energy	In fact, to equal the electrical output of a single 3.000 megawatt-electric nuclear generating station, it would take roughly 2.000 wind turbines or 140 square kilometres of solar panels.	argument

Tab. 5.1: For testing we use the UKP Sentential Argument Mining corpus [35] (top), that covers sentences to several controversial topics (bottom).

where $\boldsymbol{W}_q$, $\boldsymbol{W}_v$, and $\boldsymbol{w}_m$ are trainable weight parameters, and q_i denotes the i-th path of a token v. With the attention weights, we compute the final, weighted path vector $\boldsymbol{u}$ as:

$$\alpha_i \propto f_{attention}(\boldsymbol{q}_i, \boldsymbol{v}),$$
$$\boldsymbol{u} = \sum_{i=1}^{n} \boldsymbol{q}_i \alpha_i.$$

Vector u is then concatenated with the token embedding v and passed into the sentence encoder BiLSTM. The procedure is repeated for all tokens of the input sentence. Finally, we take the last hidden state of the sentence encoder BiLSTM as the input for a classification layer with two neurons and use a Softmax activation function to determine the class label.

5.2 Experimental Setting

The main framework, including the property selection as well as the structured and unstructured knowledge retrieval, is implemented in Python. We compare our framework to the baseline and partial improvements on eight datasets of the UKP Sentential Argument Mining Corpus (Table 5.1) [35]. We run our experiments on a single device with an Intel Core i5-10210U CPU and a 16GB RAM. Each

dataset holds a collection of sentences on one specific topic. The instances cover the following controversial topics: Abortion, cloning, death penalty, gun control, marijuana legalization, minimum wage, nuclear energy, and school uniforms. Each instance either supports or opposes a topic, or is off-topic. We train the knowledge graph embeddings for the classifier on a Wikidata dump with openKE [15]. Emphasizing robustness of the models, we test them cross-topic. Therefore, we use 70% train and 10% validation data of seven topics for training, and 20% test data of the eight unseen topic for testing. For the BiLSTM we used the following hyperparameters: $num\_seeds = 10$, $dropout = 0.7$, $lstm\_size = 64$, $batch\_size = 16$, $learning\_rate = 0.001$, $epochs = 10$, $attention\_size = 50$, $max\_paths = 10$, $max\_path\_len = 15$. We define the main core and upgrade it with the previously discussed methods for further investigations:

+WD only takes the two most frequent Wikidata properties ("subclass of", "instance of") into account to build the knowledge graph in maximum $n_d = 10$ iterations, focusing up to $k_s = 5$ concepts in the sentence and $k_t = 3$ in the topic, as our preliminary experiments have shown to return good results.

+WD+LDA uses the top $n_p = 50$ properties selected via a LDA model of 200 iterations. We set the number of topics to $n_t = 5$, the TF-IDF threshold to $t_t = 2.5$, and the minimum occurrence of a property on Wikidata to $t_c = 1000$.

+WD+LDA+GloVe includes word embeddings to avoid out-of-context parts in the graph. Focusing on low complexity, we use 50-dimensional vectors pre-trained on Wikipedia 2014 and Gigaword 5[1]. We choose a cosine similarity cutoff $t_{cos} = 0.4$ on which sample-based error analysis has shown a good performance. Thus, we build a graph covering up to $n_n = 200$ nodes.

+WD+LDA+GloVe+OpenIE enriches the aforementioned graph with unstructured data via OpenIE. From the first $n_u = 3$ websites we process between $n_{cmin} = 3000$ and $n_{cmax} = 6000$ characters to let OpenIE extract up to $n_a = 500$ annotations.

5.3 Experimental Results

Table 5.2 shows the quantitative results, meaning the number of paths and their length for each sentence, of our methods. Our main core (+WD) building the knowledge graph based on two Wikidata properties achieves the highest number of hops, which represents the overall searchable depth. However, the average path is only 1-2 nodes long, which demonstrates the ineffectiveness of the 8 hops.

[1] `https://catalog.ldc.upenn.edu/LDC2011T07`

model	label	avg #hops	avg path len	avg #sen→sen	avg #sen→top	sents with sen→sen	sents with sen→top
Main core (+WD)	argument	**8.0616**	2.2330	0.1847	1.2263	0.1547	0.5474
	no argument	7.7685	1.2444	0.1414	0.5982	0.1164	0.2553
	all	7.8943	1.6688	0.1600	0.8679	0.1329	0.3807
+LDA	argument	5.4324	4.6434	1.4457	1.9933	0.5609	0.6928
	no argument	5.0753	3.4474	1.0778	1.0552	0.4178	0.4015
	all	5.2285	3.9606	1.2357	1.4577	0.4792	0.5265
+LDA+GloVe	argument	6.6467	5.0282	1.6169	2.1459	0.5589	0.7083
	no argument	6.2837	4.0430	1.3922	1.2261	0.4588	0.4392
	all	6.4385	4.4632	1.4880	1.6184	0.5015	0.5540
+LDA+GloVe+OpenIE	argument	6.4345	7.0336	**7.9718**	**11.3184**	**0.8673**	**0.9552**
	no argument	6.3476	**7.3088**	6.0652	9.6826	0.7629	0.8984
	all	6.3850	7.1903	6.8864	10.3871	0.8079	0.9229

Tab. 5.2: Quantitative results of the proposed methods ($n_n = 600$) on 200 instances of each of the eight test sets over ten seeds. Paths between entities within the sentence are referenced as "sen→sen" and connecting the sentence with the topic as "sen→top". All numbers are relative. Bold numbers indicate the highest score in the column.

For only 15% of the sentences that are arguments to the specific topic it finds a path connecting concepts within the sentence, and for only one out of two a path connecting the sentence with the topic. Selecting the properties via a topic model (+LDA) lowers the hop count drastically, because every node now has 50 instead of 2 possible successors. The number of arguments with sentence-topic paths increases by 15% and sentence-sentence paths by over 40%. Sparsing the graph via word embeddings (+GloVe) successfully increases the hops back over 6 while keeping the relevant paths, as the other statistics slightly improve. Enriching the graph with unstructured knowledge (+OpenIE) doubles the size of the graph, so the average path length almost doubles. The number of arguments with sentence-sentence paths goes up to over 86% and with sentence-topic paths to over 95%. For both, sentences with sentence-sentence paths as well as sentences with sentence-topic paths, the margin between arguments and non-arguments stays the same for all methods except for the topic coherence in the last method. The non-arguments also provide sentence-topics paths in almost 90% of the instances. This demonstrates the relevance of both measures, sentence-sentence paths and sentence-topic paths. A sentence can be related to a topic but not provide any argumentative relevance (e.g. an opinion). Therefore, it is important to have enough training material for argument identification, which is why the overall growth from 13% to 80% in sentence-sentence paths and from 38% to 92% in sentence-topic paths is a huge success.

Evaluating the qualitative performance of our methods, Table 5.3 shows the F1 and accuracy score on each dataset and their average. On all sets the accuracy increases in every supplement, peaking in our final framework with scores between 81%

model	dataset	accuracy	F1
+WD	abortion	0.7592 ± 0.0137	0.6532 ± 0.0093
	cloning	0.7825 ± 0.0092	0.6607 ± 0.0183
	death penalty	0.8137 ± 0.0101	0.6172 ± 0.0178
	gun control	0.7913 ± 0.0067	0.7496 ± 0.0104
	marijuana legalization	0.7898 ± 0.0071	0.6912 ± 0.0102
	minimum wage	0.7724 ± 0.0067	0.5555 ± 0.0209
	nuclear energy	0.7566 ± 0.0090	0.7102 ± 0.0165
	school uniforms	0.7878 ± 0.0132	0.4493 ± 0.0264
	average	0.7817 ± 0.0095	0.6359 ± 0.0162
+WD+LDA	abortion	0.7998 ± 0.0010	0.6880 ± 0.0215
	cloning	0.7980 ± 0.0118	0.6393 ± 0.0128
	death penalty	0.7903 ± 0.0060	0.6275 ± 0.0236
	gun control	0.7916 ± 0.0094	0.6674 ± 0.0350
	marijuana legalization	0.8024 ± 0.0095	**0.7778** ± 0.0249
	minimum wage	0.7870 ± 0.0080	0.7451 ± 0.0186
	nuclear energy	0.8022 ± 0.0102	0.6045 ± 0.0308
	school uniforms	0.7974 ± 0.0044	0.5201 ± 0.0142
	average	0.7961 ± 0.0095	0.6587 ± 0.0227
+WD+LDA+GloVe	abortion	0.8139 ± 0.0080	0.7111 ± 0.0310
	cloning	0.8107 ± 0.0173	0.6432 ± 0.0406
	death penalty	0.8212 ± 0.0130	0.5666 ± 0.0341
	gun control	0.8170 ± 0.0168	0.5959 ± 0.0524
	marijuana legalization	0.8033 ± 0.0207	0.6384 ± 0.0388
	minimum wage	0.8003 ± 0.0276	0.7573 ± 0.0340
	nuclear energy	0.8295 ± 0.0154	0.5817 ± 0.0317
	school uniforms	0.8341 ± 0.0029	0.4878 ± 0.0321
	average	0.8163 ± 0.0152	0.6227 ± 0.0368
+WD+LDA+GloVe+OpenIE	abortion	0.8726 ± 0.0116	0.6916 ± 0.0154
	cloning	0.8500 ± 0.0021	0.7224 ± 0.0381
	death penalty	0.8630 ± 0.0031	0.6415 ± 0.0302
	gun control	0.8542 ± 0.0097	0.7476 ± 0.0025
	marijuana legalization	0.8225 ± 0.0232	0.6300 ± 0.0159
	minimum wage	0.8150 ± 0.0116	0.7712 ± 0.0227
	nuclear energy	0.8500 ± 0.0021	0.7045 ± 0.0285
	school uniforms	**0.8760** ± 0.0202	0.5194 ± 0.0139
	average	0.8504 ± 0.0104	0.6785 ± 0.0214

Tab. 5.3: Accuracy and F1 of the proposed methods on 200 instances of each of the eight test sets over ten seeds. In each model, the graph built upon structured knowledge from Wikidata covers up to $n_n = 600$ nodes. Bold numbers indicate the highest score in the column, underlined numbers indicate the highest score on this dataset.

model	dataset	accuracy	F1
Baseline	average		0.6069 ± 0.0074
Our framework	abortion	0.8398 ± 0.0017	**0.6993** ± 0.0124
	cloning	0.8437 ± 0.0019	0.6844 ± 0.0129
	death penalty	**0.8502** ± 0.0051	0.6677 ± 0.0124
	gun control	0.8392 ± 0.0022	0.6528 ± 0.0283
	marijuana legalization	0.8472 ± 0.0015	0.6749 ± 0.0196
	minimum wage	0.8412 ± 0.0013	0.6907 ± 0.0170
	nuclear energy	0.8374 ± 0.0032	0.6328 ± 0.0310
	school uniforms	0.8343 ± 0.0011	0.5451 ± 0.0198
	average	0.8416 ± 0.0023	0.6560 ± 0.0192

Tab. 5.4: Accuracy and F1 of the baseline versus our framework ($n_n = 200$) on the UKP Sentential Argument Mining Corpus over ten seeds. Bold numbers indicate the highest score in the column.

("minimum wage") and 87% ("school uniforms"). Our framework also performs best on F1 score, improving on the main core by 4%. Regarding F1 only four datasets peak in the final framework, which shows that unstructured knowledge is not helpful in every context. Two datasets ("gun control", and "nuclear energy") even perform best in our main core demonstrating the relevance of the topic. Some types of context might be very specific and cover lots of technical terms that external knowledge sources might not be able to model well. Overall our methods show great improvements on both measures, increasing on accuracy by 7% and on F1 by 4%.

Testing our final framework on the complete UKP Sentential Argument Mining Corpus, Table 5.4 shows our performance against the baseline. Seven out of eight topics beat the average F1 of the baseline, with the "abortion"-dataset being the peak at 69.93% on F1. Overall, our framework beats the baseline by 5% on F1, achieving 84.16% on accuracy and 65.60% on F1. Thus, we show an improvement on argument identification with the help of our methods.

Studying the complexity performance of our methods, Figure 5.1 show the average runtime of our methods. +LDA (third bar) increases the runtime from 30 to 276 seconds because the number of edges per node went from 2 to 50. +Glove (fourth bar) calculates a sentence vector and the cosine similarity of each word vector and the sentence vector, pushing the runtime to 301 seconds. +OpenIE (fifth bar) works with the Java framework OpenIE which, embedding it into a Python framework, consumes time. Combined with the Google search via a HTTP request, the runtime peaks at 334 seconds. Focusing real world applications we lower the number of nodes of the knowledge graoh from 600 to 200 (red bar), resulting in a third of the original runtime of our framework. On average, every instances now is executed in 121 seconds.

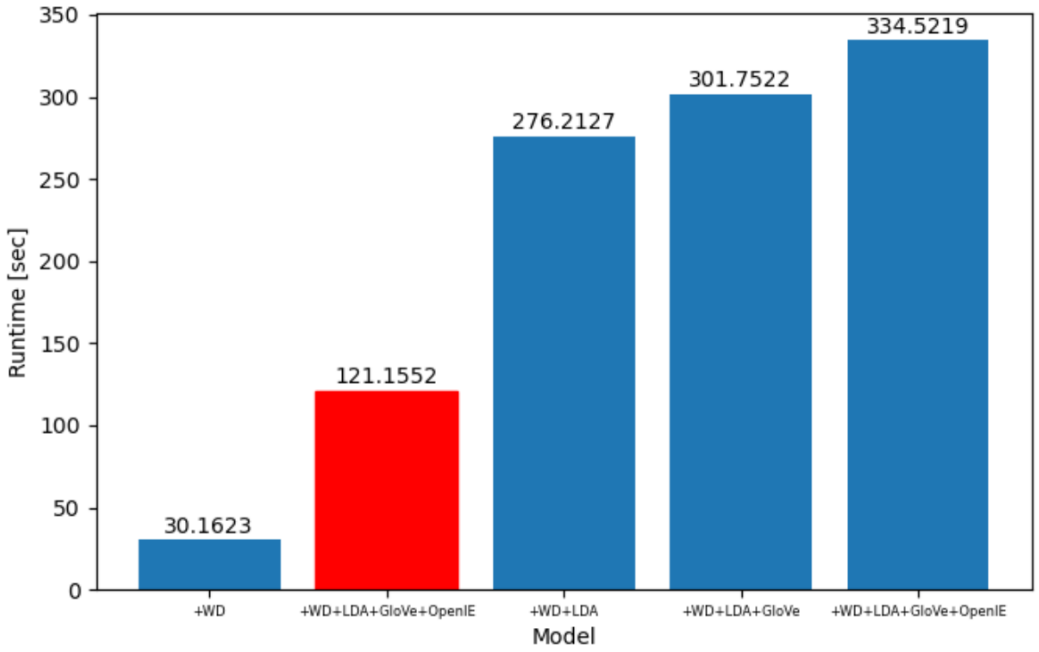


Fig. 5.1: Average runtime per instance of our proposed methods on 200 instances of each of the eight test sets. Methods with blue bars build a structured-knowledge based graph covering up to $n_n = 600$ nodes, compared to our final framework with up to $n_n = 200$ nodes in red.

5.3.1 Error Analysis

To investigate under what conditions our framework finds irrelevant paths, we conduct a qualitative error analysis. We randomly selected error paths which were found by our framework. The reasons are mainly (1) questionable entity linking by Wikifier, (2) falsely connecting nodes from the two graphs, and (3) noisy data. The first example in Table 5.5 generates the following path:

$$\textit{There is} \xrightarrow{WIKIFIERED} \textit{English grammar}\ (Q560583) \xrightarrow[P361]{\textit{part of}} \textit{English}\ (Q1860) \xrightarrow[P17]{\textit{country}} \textit{United States of America}\ (Q30) \xrightarrow{MATCH} \textit{state} \xrightarrow{\textit{considered}} \textit{abortion} \xrightarrow{MATCH} \textit{abortion}\,.$$

Due to the Wikifier mapping *There is* to the Wikidata entity *English grammar* (Q560583) this path falsely connects the sentence to the topic. This problem can be mitigated by specifying the Wikifier parameters less error-prone. The second example in Table 5.5 produces the path:

$$\textit{hate} \xrightarrow{WIKIFIERED} \textit{Hate speech}\ (Q653347) \xrightarrow[P5008]{\textit{on focus list of Wikimedia project}} \textit{WikiForHumanRights}\ (Q78499962) \xrightarrow[P31]{\textit{instance of}} \textit{human rights}\ (Q8458) \xrightarrow{MATCH} \textit{humanity} \xrightarrow{has} \textit{energy needs} \xrightarrow{MATCH} \textit{energy}\,.$$

topic	sentence	label
Abortion	There is no third possibility.	no-argument
Nuclear energy	We hate spam too!	no-argument
School uniforms	Wearing school uniform is also important for you to save time changing clothes or outfits because you are trying to figure out the right combination of shirt, pants, blouse or skirt.	argument
Gun control	While this danger is likely to endure, the easy availability of firearms makes such hate crimes far more deadly and far more frightening.	argument

Tab. 5.5: Randomly sampled sentences from the testing corpus.

Although every edge in the path makes some sense, the total connection of *hate* and *energy* via the context of human rights is rather questionable. This problem can be mitigated by applying further cleaning strategies on the used knowledge.

5.3.2 Case Study

The example in Figure 4.7 demonstrates the power of our method. Calculating the sentence vector v_s and looking up the entity vectors yields:

$$\begin{aligned} cos(v_s, GV(\text{“office”})) &\approx 0.7007 \\ cos(v_s, GV(\text{“room”})) &\approx 0.7195 \\ cos(v_s, GV(\text{“location”})) &\approx 0.6469 \\ cos(v_s, GV(\text{“space”})) &\approx 0.6210 \\ cos(v_s, GV(\text{“spacetime”})) &\approx -0.0365 \\ cos(v_s, GV(\text{“time”})) &\approx 0.8891. \end{aligned}$$

Since the offices-related concepts and times-related concepts are relevant to this specific sentence, it makes sense that the vectors of corresponding entities have a cosine similarity close to 1. Whereas the entity spacetime might be related to both space and time, it is not relevant to this sentence. Therefore, cutting at the threshold of $t_{cos} = 0.4$ utilizes this fact and prevents from finding this path.

Furthermore, we randomly picked correct classified instances and now study the paths found by our framework. The third example in Table 5.5 creates the following paths:

$$\text{school} \xrightarrow{MATCH} school\ students \xrightarrow{wear\ name\ tags\ on} hand\ side\ of\ their\ shirts \xrightarrow{MATCH} Shirt\ (Q76768) \xrightarrow[P527]{has\ part} Blouse\ (Q152563) \xrightarrow{WIKIFIERED} blouse\,,$$

school $\xrightarrow{MATCH}$ *High school girls* $\xrightarrow{must\ wear}$ *skirts* $\xrightarrow{MATCH}$ *skirt* ,

school $\xrightarrow{MATCH}$ *school students* $\xrightarrow{wear\ name\ tags\ on}$ *hand side of their shirts* $\xrightarrow{MATCH}$ *shirt* ,

school $\xrightarrow{MATCH}$ *school boys* $\xrightarrow{wear}$ *trousers* $\xrightarrow{MATCH}$ *Trousers* (*Q*39908) $\xrightarrow{WIKIFIERED}$ *pants* ,

skirt $\xrightarrow{MATCH}$ *Their skirts* $\xrightarrow{vary\ depending}$ *depending school* $\xrightarrow{MATCH}$ school ,

school uniform $\xrightarrow{WIKIFIERED}$ *School uniform* (*Q*862633) $\xrightarrow{WIKIFIERED}$ *school uniforms* ,

shirt $\xrightarrow{WIKIFIERED}$ *Shirt* (*Q*76768) $\xrightarrow[P527]{has\ part}$ *Blouse* (*Q*152563) $\xrightarrow{WIKIFIERED}$ *blouse* .

For the fourth example in Table 5.5, our framework creates the following paths:

crimes $\xrightarrow{WIKIFIERED}$ *Crime* (*Q*83267) $\xrightarrow[P279]{subclass\ of}$ *statutorylaw* (*Q*7766927) $\xrightarrow{MATCH}$ *laws* $\xrightarrow{Prevention\ of}$ *firearm − related injuries* $\xrightarrow{MATCH}$ *firearms* ,

crimes $\xrightarrow{WIKIFIERED}$ *Crime* (*Q*83267) $\xrightarrow[P279]{subclass\ of}$ *statutory law* (*Q*7766927) $\xrightarrow{MATCH}$ *stand − your − ground laws* $\xrightarrow{may\ increase}$ *violent crime* $\xrightarrow{MATCH}$ *Hate crime* (*Q*459409) $\xrightarrow{WIKIFIERED}$ *hate* ,

hate $\xrightarrow{WIKIFIERED}$ *Hatecrime* (*Q*459409) $\xrightarrow[P279]{subclass\ of}$ *Crime* (*Q*83267) $\xrightarrow{WIKIFIERED}$ *crimes* ,

crimes $\xrightarrow{WIKIFIERED}$ *Crime* (*Q*83267) $\xrightarrow[P279]{subclass\ of}$ *statutory law* (*Q*7766927) $\xrightarrow{MATCH}$ *laws* $\xrightarrow{requiring}$ *people* $\xrightarrow{owning}$ *gun* $\xrightarrow{MATCH}$ gun ,

gun control $\xrightarrow{MATCH}$ *guns* $\xrightarrow{MATCH}$ *gun* $\xrightarrow{WIKIFIERED}$ *Handgun*(*Q*1574963) $\xrightarrow[P279]{subclass\ of}$ *Firearm* (*Q*12796) $\xrightarrow{WIKIFIERED}$ *firearms* ,

gun control $\xrightarrow{MATCH}$ *guns* $\xrightarrow{recovered\ in}$ *crimes of states in U.S.* $\xrightarrow{MATCH}$ *Hate crime* (*Q*459409) $\xrightarrow{WIKIFIERED}$ *hate* .

In both examples, the paths successfully connect several concepts found in the topics and the sentences. It is unclear whether the paths represent the actual essence of the arguments, although they provide useful background knowledge how the concepts interact and therefore prove some level of sense and topic-coherence.

6 Conclusion

We presented a novel argument mining framework, which improves knowledge extraction using structured and unstructured data. We overcome the problem of exponential growth of the knowledge graph needed for path extraction using two key ideas: topic modeling to improve the selection of relevant properties, and word embedding to ensure topical consistency, which leads to a sparser knowledge graph. In comparison to existing methods, we can process a much larger amount of data and take more possible properties into consideration. This allows us to discover more relevant paths. Our results show an average performance of 84.16% for the accuracy and 65.60% for the F1 measure on the UKP Sentential Argument Mining Corpus. This may contribute to future work that combines argument mining with discrete knowledge in the form of knowledge graphs.

6.1 Future work

As our experiments show, parameters like the graph size have to be chosen wisely depending on the given application. Often times it is a trade-off between performance and complexity. Especially the topic-wise qualitative results demonstrate the power of a topic. To emphasize the topic relevance of a given argument, future work might supplement a main core with a selection of methods depending on the complexity and specificity of the given topic. Speaking of real-world applications, parallelization and distribution play a huge role. Future work might therefore dump a version of Wikidata to overcome SparQL limitations and parallelize the dynamic BFS on a local Wikidata graph.

Bibliography

[1] Yamen Ajjour, Wei-Fan Chen, Johannes Kiesel, Henning Wachsmuth, and Benno Stein. „Unit Segmentation of Argumentative Texts“. In: *Proceedings of the 4th Workshop on Argument Mining*. 2017, 118–128 (cit. on p. 18).

[2] Gabor Angeli, Melvin Jose Johnson Premkumar, and Christopher D. Manning. „Leveraging Linguistic Structure For Open Domain Information Extraction“. In: *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Beijing, China: Association for Computational Linguistics, July 2015, pp. 344–354 (cit. on p. 15).

[3] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. „Latent Dirichlet Allocation“. In: *J. Mach. Learn. Res.* 3.null (Mar. 2003), 993–1022 (cit. on pp. 1, 14).

[4] Teresa Botschen, Daniil Sorokin, and Iryna Gurevych. „Frame- and Entity-Based Knowledge for Common-Sense Argumentative Reasoning“. In: *Proceedings of the 5th Workshop on Argument Mining*. Brussels, Belgium: Association for Computational Linguistics, Nov. 2018, pp. 90–96 (cit. on pp. 1, 5, 6).

[5] Janez Brank, Gregor Leban, and Marko Grobelnik. „Annotating documents with relevant Wikipedia concepts“. In: *Proceedings of Slovenian KDD Conference on Data Mining and Data Warehouses (SiKDD)*. 2017 (cit. on p. 19).

[6] Daniel Cer, Yinfei Yang, Sheng yi Kong, et al. *Universal Sentence Encoder*. 2018. arXiv: 1803.11175 [cs.CL] (cit. on p. 12).

[7] Peter Clark, Oren Etzioni, Daniel Khashabi, et al. *From 'F' to 'A' on the N.Y. Regents Science Exams: An Overview of the Aristo Project*. 2019. arXiv: 1909.01958 [cs.CL] (cit. on pp. 1, 7, 29).

[8] Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. „Supervised Learning of Universal Sentence Representations from Natural Language Inference Data“. In: *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing* (2017) (cit. on p. 12).

[9] Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. „Indexing by latent semantic analysis“. In: *Journal of the American Society for Information Science* 41.6 (1990), pp. 391–407. eprint: https://asistdl.onlinelibrary.wiley.com/doi/pdf/10.1002/%28SICI%291097-4571%28199009%2941%3A6%3C391%3A%3AAID-ASI1%3E3.0.CO%3B2-9 (cit. on p. 12).

[10]Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. 2018. arXiv: 1810.04805 [cs.CL] (cit. on pp. 1, 8).

[11]Steffen Eger, Johannes Daxenberger, and Iryna Gurevych. „Neural End-to-End Learning for Computational Argumentation Mining“. In: *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Vancouver, Canada: Association for Computational Linguistics, July 2017, pp. 11–22 (cit. on p. 18).

[12]Michael Färber, Basil Ell, and Carsten Menne. „A Comparative Survey of DBpedia , Freebase , OpenCyc , Wikidata , and YAGO“. In: 2015 (cit. on p. 17).

[13]Michael Fromm, Evgeniy Faerman, and Thomas Seidl. „TACAM: Topic And Context Aware Argument Mining“. In: *IEEE/WIC/ACM International Conference on Web Intelligence*. WI '19. New York, NY, USA: Association for Computing Machinery, 2019, 99–106 (cit. on pp. 1, 6).

[14]Jian Guan, Fei Huang, Zhihao Zhao, Xiaoyan Zhu, and Minlie Huang. *A Knowledge-Enhanced Pretraining Model for Commonsense Story Generation*. 2020. arXiv: 2001.05139 [cs.CL] (cit. on pp. 1, 8).

[15]Xu Han, Shulin Cao, Xin Lv, et al. „OpenKE: An Open Toolkit for Knowledge Embedding“. In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Nov. 2018, pp. 139–144 (cit. on p. 35).

[16]Bin He, Di Zhou, Jinghui Xiao, et al. *Integrating Graph Contextualized Knowledge into Pre-trained Language Models*. 2019. arXiv: 1912.00147 [cs.CL] (cit. on pp. 1, 8).

[17]Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, et al. „Teaching Machines to Read and Comprehend“. In: *Advances in Neural Information Processing Systems*. 2015, pp. 1693–1701 (cit. on p. 33).

[18]Sepp Hochreiter and Jürgen Schmidhuber. „Long Short-Term Memory“. In: *Neural Comput.* 9.8 (Nov. 1997), 1735–1780 (cit. on p. 33).

[19]Ryan Kiros, Yukun Zhu, Ruslan Salakhutdinov, et al. *Skip-Thought Vectors*. 2015. arXiv: 1506.06726 [cs.CL] (cit. on p. 12).

[20]Anne Lauscher, Ivan Vulic, Edoardo Maria Ponti, Anna Korhonen, and Goran Glavas. „Informing Unsupervised Pretraining with External Linguistic Knowledge“. In: *ArXiv* abs/1909.02339 (2019) (cit. on pp. 1, 8).

[21]Yoav Levine, Barak Lenz, Or Dagan, et al. „SenseBERT: Driving Some Sense into BERT“. In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics, July 2020 (cit. on pp. 1, 8).

[22]Ran Levy, Yonatan Bilu, Daniel Hershcovich, Ehud Aharoni, and Noam Slonim. „Context Dependent Claim Detection“. In: *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*. Dublin, Ireland: Dublin City University and Association for Computational Linguistics, Aug. 2014, pp. 1489–1500 (cit. on p. 18).

[23]Shangwen Lv, Daya Guo, Jingjing Xu, et al. „Graph-Based Reasoning over Heterogeneous External Knowledge for Commonsense Question Answering“. In: *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*. AAAI Press, 2020, pp. 8449–8456 (cit. on pp. 1, 7, 8, 29).

[24]Christopher D. Manning, Mihai Surdeanu, John Bauer, et al. „The Stanford CoreNLP Natural Language Processing Toolkit“. In: *Association for Computational Linguistics (ACL) System Demonstrations*. 2014, pp. 55–60 (cit. on pp. 8, 29).

[25]Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. „Linguistic Regularities in Continuous Space Word Representations“. In: *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Atlanta, Georgia: Association for Computational Linguistics, June 2013, pp. 746–751 (cit. on p. 12).

[26]Saif Mohammad, Svetlana Kiritchenko, Parinaz Sobhani, Xiaodan Zhu, and Colin Cherry. „SemEval-2016 Task 6: Detecting Stance in Tweets“. In: *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*. San Diego, California: Association for Computational Linguistics, June 2016, pp. 31–41 (cit. on p. 18).

[27]Malte Ostendorff, Peter Bourgonje, Maria Berger, et al. *Enriching BERT with Knowledge Graph Embeddings for Document Classification*. 2019. arXiv: `1909.08402 [cs.CL]` (cit. on pp. 1, 7, 8).

[28]Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. *The PageRank Citation Ranking: Bringing Order to the Web*. Technical Report 1999-66. Previous number = SIDL-WP-1999-0120. Stanford InfoLab, Nov. 1999 (cit. on pp. 7, 8).

[29]Jeffrey Pennington, Richard Socher, and Christopher D. Manning. „GloVe: Global Vectors for Word Representation“. In: *Empirical Methods in Natural Language Processing (EMNLP)*. 2014, pp. 1532–1543 (cit. on p. 12).

[30]Matthew Peters, Mark Neumann, Mohit Iyyer, et al. „Deep Contextualized Word Representations“. In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)* (2018) (cit. on pp. 8, 12).

[31]Matthew E. Peters, Mark Neumann, Robert Logan, et al. „Knowledge Enhanced Contextual Word Representations“. In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 43–54 (cit. on pp. 1, 8).

[32]Peter Potash, Robin Bhattacharya, and Anna Rumshisky. „Length, Interchangeability, and External Knowledge: Observations from Predicting Argument Convincingness“. In: *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Taipei, Taiwan: Asian Federation of Natural Language Processing, Nov. 2017, pp. 342–351 (cit. on pp. 1, 5, 6, 29).

[33]Ruty Rinott, Lena Dankin, Carlos Alzate Perez, et al. „Show Me Your Evidence - an Automatic Method for Context Dependent Evidence Detection“. In: *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Lisbon, Portugal: Association for Computational Linguistics, Sept. 2015, pp. 440–450 (cit. on p. 18).

[34] Christian Stab, Johannes Daxenberger, Chris Stahlhut, et al. „ArgumenText: Searching for Arguments in Heterogeneous Sources“. In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*. New Orleans, Louisiana: Association for Computational Linguistics, June 2018, pp. 21–25 (cit. on p. 2).

[35] Christian Stab, Tristan Miller, Benjamin Schiller, Pranav Rai, and Iryna Gurevych. „Cross-topic Argument Mining from Heterogeneous Sources“. In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Brussels, Belgium: Association for Computational Linguistics, Nov. 2018, pp. 3664–3674 (cit. on pp. 1, 3, 17, 18, 34).

[36] Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. In: *Proceedings of the 2019 Conference of the North* (2019) (cit. on p. 8).

[37] Henning Wachsmuth, Martin Potthast, Khalid Al-Khatib, et al. „Building an Argument Search Engine for the Web“. In: *Proceedings of the 4th Workshop on Argument Mining*. 2017, 49–59 (cit. on p. 18).

[38] Ruize Wang, Duyu Tang, Nan Duan, et al. *K-Adapter: Infusing Knowledge into Pre-Trained Models with Adapters*. 2020. arXiv: 2002.01808 [cs.CL] (cit. on pp. 1, 8).

[39] Wenhan Xiong, Jingfei Du, William Yang Wang, and Veselin Stoyanov. *Pretrained Encyclopedia: Weakly Supervised Knowledge-Pretrained Language Model*. 2019. arXiv: 1912.09637 [cs.CL] (cit. on pp. 1, 8).

[40] Zhengyan Zhang, Xu Han, Zhiyuan Liu, et al. „ERNIE: Enhanced Language Representation with Informative Entities“. In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, July 2019, pp. 1441–1451 (cit. on pp. 1, 8).

List of Figures

List of Tables

Declaration

I hereby declare that I have written the present thesis independently and without use of other than the indicated means. I also declare that to the best of my knowledge all passages taken from published and unpublished sources have been referenced. The paper has not been submitted for evaluation to any other examining authority nor has it been published in any form whatsoever.

Mainz, 27.10.2020

Patrick Abels