Topic-Guided Knowledge Graph Construction for Argument Mining

Weichen Li[1*], Patrick Abels[2*], Zahra Ahmadi[3], Sophie Burkhardt[1], Benjamin Schiller[4],
Iryna Gurevych[4], Stefan Kramer[2]
[1]*Technical University of Kaiserslautern*, Kaiserslautern, Germany
[2]*Johannes Gutenberg University Mainz*, Mainz, Germany
[3]*L3S Research Center, Leibniz University Hannover*, Hannover, Germany
[4]*Technical University of Darmstadt*, Darmstadt, Germany
{weichen,burkhardt}@cs.uni-kl.de, pabels@students.uni-mainz.de, ahmadi@l3s.de,
{schiller,gurevych}@ukp.informatik.tu-darmstadt.de, kramer@informatik.uni-mainz.de
∗ equal contribution

***Abstract*—Decision-making tasks usually follow five steps: identifying the problem, collecting data, extracting evidence, identifying arguments, and making the decision. This paper focuses on two steps of decision-making: extracting evidence by building knowledge graphs (KGs) of specialized topics and identifying sentences' arguments through sentence-level argument mining. We present a hybrid model that combines topic modeling using latent Dirichlet allocation (LDA) and word embeddings to obtain external knowledge from structured and unstructured data. We use a topic model to extract topic- and sentence-specific evidence from the structured knowledge base Wikidata. A knowledge graph is constructed based on the cosine similarity between the entity word vectors of Wikidata and the vector of the given sentence. A second graph based on topic-specific articles found via Google supplements the general incompleteness of the structured knowledge base. Combining these graphs, we obtain a graph-based model that, as our evaluation shows, successfully capitalizes on both structured and unstructured data.**

***Index Terms*—Topic model, knowledge graph, argument mining**

I. Introduction

Knowledge graphs (KGs) have been used in many real-world applications in recent years where knowledge is required to solve a task in addition to raw data. We focus on solving argument mining tasks. It is challenging to understand arguments without further relevant world knowledge. We tackle this problem by connecting the entities in a sentence and the sentence topic with paths through a locally created knowledge graph. To do this, we use structured data (Wikidata) and unstructured data (Google search) for each sentence and extract paths leading from one entity to another. Latent Dirichlet allocation (LDA) [1] improves the quality of connections in the knowledge graph by ensuring that the paths do not leave the context of the topic.

Early work introduced world knowledge into NLP tasks by harnessing manually constructed knowledge bases like Wikidata or DBPedia [2], [3]. Soon after realizing the incompleteness of structured knowledge sources, authors provided world knowledge from data such as Wikipedia [4]. Having problems with noise in unstructured sources, more recent work combines both structured and unstructured data [5]–[7].

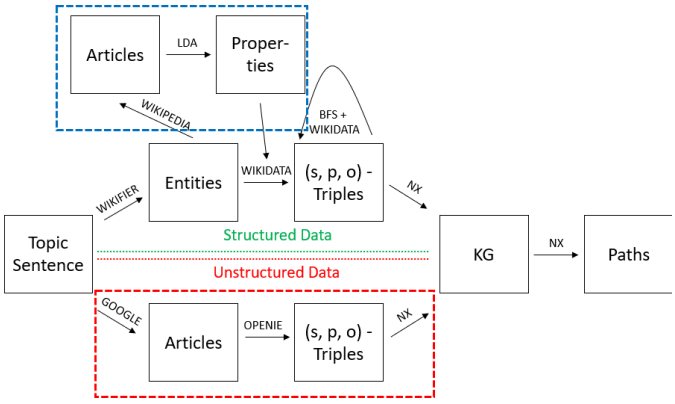

Fig. 1. The proposed framework to build Knowledge Graph paths: The main idea of using KG paths is augmented by LDA-driven property selection (blue box) and unstructured knowledge enrichment via Google search (red box).

Motivated by the similarity of argument-based tasks, utilizing and enhancing pre-trained language models like BERT [8] has gained substantial popularity [9]–[16]. In comparison, besides being simpler, our approach focuses on the knowledge graph itself and how it can be efficiently traversed. Figure 1 shows the idea of the framework. We use LDA to extract KG paths from structured and unstructured data that stay on topic. Our contributions are as follows:

1) We develop a graph-based approach leveraging evidence from structured knowledge bases via latent Dirichlet allocation.
2) We propose a comparably efficient 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 evaluate the quality of our knowledge graph on the argument mining task.

II. Related Work

Recent work that combines both structured and unstructured data proposes new benchmarks on various tasks, including

question answering [5], [6], document classification [7], and argument classification task [17] [18]. Clark *et al.* [5] 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 [19], and (3) large-scale language model methods such as ELMo [20] and BERT [8].

In a similar graph-based approach but on a different problem, Lv *et al.* [6] build two graphs on structured data from ConceptNet and unstructured data from Wikipedia for the Commonsense Question Answering problem, achieving an accuracy of 75.3% on the CommonsenseQA dataset [21]. While they select the top 10 sentences regarding the given query as Wikipedia evidence via the Elastic Search engine, we rely on several hundred of topic-specific sentences from Google search ranked via PageRank [22]. Furthermore, instead of ConceptNet, we use Wikidata as a source of structured data; hence, we use the knowledge from Wikipedia. Additionally, their rules for two nodes of the knowledge graphs being connected are relatively strict: Either one is contained in the other or only differs 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 apply *topology sort* to avoid cycles, we consider entities as vertices and relations as edges, having no need for sorting. With their restrictive search, Lv *et al.* consider 20 nodes in structured data and ten sentences from unstructured data, while both of our graphs from Wikidata and Google cover up to several hundred nodes.

Ostendorff *et al.* [7] enrich BERT with the author information via Wikidata to improve book classification. However, they do not 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.

K-BERT [23] introduces a new approach of incorporating KGs into BERT pre-training. BERT uses large-scale pre-training corpora; however, it lacks knowledge of specialized domains. Using KGs as additional information for representation learning can improve understanding of a domain and enrich the sentence information. Additionally, the outputs from pre-training can be fine-tuned on an NLP task to produce predictions. One of K-BERT's unique features is a *"visible matrix"*. The input sentences are seen as sentence trees, and each word is extended as a triple (word, relations, word2) if it exists in the provided knowledge graph. The addition of a *"seeing layer"* – another feature unique to K-BERT – then reduces the risk of knowledge noise, since the irrelevant entities cannot "visit" each other. Another advantage of K-BERT is the lower GPU memory requirement than the joint representations of the word and KG embedding. In another approach, Lausher *et al.* [24] introduced a model using *"adapter training"* to inject conceptual knowledge into BERT.

III. Method

A. Method Overview

An argument is a combination of a topic and a sentence holding evidence towards this topic, which are given as input to the following methods. Our goal is to extract relevant paths from the knowledge graph that may help in the downstream argument mining task. The paths connect entites from the given topic and sentence. As shown in Figure 1, a sentence and its topic are the inputs to the Wikifier [25][1], 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.

Our main source of knowledge is the collaboratively constructed knowledge-based Wikidata. Wikidata is free, multilingual, and its broad community curation ensures a high data quality with currently more than 88 million items E and 5519 properties R. Beyond a label (e.g., "Douglas Adams") and an identifier (e.g., "Q42"), each item covers a set of statements S linking to other items. A statement expressing semantic or ontological connections can be described as a binary relation between entities (e.g., P69(Q42, Q691283) represents 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 $S = \{r(e_1, e_2)|r \in R, e_1, e_2 \in E\}$. Each entity (e.g., "Douglas Adams (Q42)") is connected to several other entities via specific properties (e.g. "educated at (P69)"), creating a relation-based knowledge structure. Therefore, it can be represented as a list of binary relations (e.g., "educated at" ("Douglas Adams", "St John's College")). Graphically, this list can also be seen as a list of triples, with each entity being a node and each relation being an edge connecting these two nodes. Starting from this graph, in every n_d iterations, it queries Wikidata via SPARQL-queries for a list of entities connected to one of the unseen nodes in the graph via one of the properties of interest. Figure 2 represents a sample Wikidata query with its returned list of object-entities. By 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 are added as additional knowledge to help classify whether the given sentence is an argument to the given topic.

[1] http://wikifier.org.

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

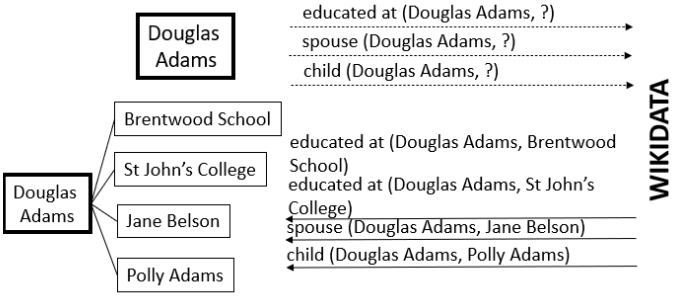

Fig. 2. Expansion of the node "Douglas Adams" via a SPARQL query (dashed arrow) per entity per property.

B. Property Selection with Latent Dirichlet Allocation

Building a knowledge graph over all the existing properties per node is infeasible as: (1) In the worst case, at depth D the graph already has $\sum_{d=0}^{D} E_0 P^D$ nodes and $\sum_{d=1}^{D} E_0 P^D$ edges with E_0 being the number of nodes that we started with and P the number of properties. With the BFS runtime of a graph being $\mathcal{O}(|edges| + |vertices|)$, the search is not feasible for P=5519. (2) Most of the properties do not help gain 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 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").

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

1) Load the corresponding Wikipedia articles using the Wikipedia-API to find the given entities' articles and extract the relevant texts (lines 1-3).
2) Train an LDA model to extract the most relevant topics for this specific context (line 4). Preprocessing steps include removing stopwords, punctuations, and the numeric expressions from the articles after converting them to lowercase, and finally, applying tokenization and lemmatization. We build a corpus based on the preprocessed text, which we apply LDA on. Then, we extract the n_t best topics represented as a batch of words each (line 5).

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

1 $D \leftarrow \emptyset$
2 **forall** $e \in E$ **do**
3 $\quad D \leftarrow D \cup$ Wikipedia article of e
4 $L \leftarrow$ LDA(D, n_t)
5 $T \leftarrow$ top-topics(L)
6 $W \leftarrow \emptyset$
7 **forall** $t \in T$ **do**
8 $\quad$ **forall** $w \in t$ **do**
9 $\quad\quad$ **if** $w \notin W$ ***and*** $w \notin$ *stopwords* **then**
10 $\quad\quad\quad W \leftarrow W \cup w$
11 $F \leftarrow$ *RANK_BY_TFIDF*(W, t_t)
12 $\tilde{P} \leftarrow \emptyset$
13 **forall** $w \in F$ **do**
14 $\quad$ **forall** $p \in P$ **do**
15 $\quad\quad$ **if** $p \notin \tilde{P}$ ***and*** $w \in p.info$ ***and*** $p.count > t_c$ **then**
16 $\quad\quad\quad \tilde{P} \leftarrow \tilde{P} \cup p$
17 **return** *SELECT_FREQUENT*$(\tilde{P}, n_p)$

3) Rank the words representing the topics by their relevance to the properties measured via TF-IDF (lines 6-11). 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 and rank the words by their cumulative score

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

proceeding with only those achieving a given threshold.
4) Extract top-related properties by searching the property descriptions for the top-ranked words (lines 12-16). We consider every property with at least one word appearing at least once. Ranked by their number of appearances (line 17), we return the n_p most frequent properties.

C. Knowledge Retrieval with Dynamic BFS and Word Embedding

Only one node connection out of the context can result in a noisy path when building a graph dynamically. 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 a diverse graph is not optimal; instead, a more profound exploration of one relevant subgraph could yield important information. We address these challenges by only expanding child nodes if the GloVe word embedding [26] of every node (entity vector) belongs to the context of the input sentence

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

1 $G \leftarrow$ MultiDiGraph()
2 **for** $i = 0 \dots \#E - 1$ **do**
3 $\quad G \leftarrow G \cup edge(C_i, \text{“WIKIFIERED”}, E_i)$
4 $v_s \leftarrow \text{avg}(\{GV(t) | t \in \text{tokenize}(\text{lower}(s))\})$
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_SIM*$(GV, v_s, e_{new}, t_{cos})$ **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

(sentence vector) with respect to a cosine similarity threshold. GloVe primarily performs well on word analogy and word similarity tasks, making it suitable for 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.

We map each word to its GloVe embedding and propose an augmentation method to any BFS-based graph construction in Algorithm 2. We initialize a directional graph and allow multiple edges between two nodes via NetworkX[3]. For every concept in the given topic or sentence annotated by an entity via the Wikifier, we add two nodes and an 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 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 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

[3]https://networkx.github.io/.

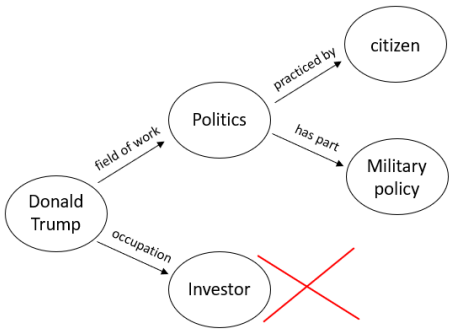

Fig. 3. Visualization of how word embeddings make the graph more sparse.

entity e_{new} to the graph, we prove it to fulfill one of the following conditions:

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

where t_{cos} is a threshold and v_t is the GV-vectors of the lowercase tokens in e_{new}. We thereby assure that at least one of the tokens in the new entity stays in the given sentence context. 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 3 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:

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

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

D. Enriching the Knowledge graph with OpenIE

Recent studies proved the importance of additional unstructured data to compensate for the incompleteness of structured knowledge bases like Wikidata [4]–[6]. However, most approaches do not pick the relevant data and/or handle the noise. Instead of using either hand-picked or randomly chosen articles [4], we consider top-ranked articles based on PageRank in the Google searches for a given topic. Using OpenIE [19], we build a second knowledge graph and combine them gradually to handle the noise. We enrich the knowledge graph with unstructured data in 4 steps (Algorithm 3):

1) By searching Google for a given topic, we receive a list of websites ranked by Google’s PageRank algorithm (line 1). Considering the top n_u websites, we extract each sentence with at least three words (line 5) – the minimum to let OpenIE build a triple from. To improve efficiency (runtime, storage, and OpenIE limits), we rank the sentences by their total character length (line 6) and

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 $n_{c_{min}}$

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 $G \leftarrow G +$ *edge*(subject, predicate, object)
12 **forall** $v \in V_G$ **do**
13 **if** *MATCH*$(v,$ *subject*$)$ **then**
14 $G \leftarrow G \cup$ *edge*$(v,$ "OPENIED", subject$)$
15 **if** *MATCH*$(v,$ *object*$)$ **then**
16 $G \leftarrow G \cup$ *edge*$(v,$ "OPENIED", object$)$
17 **return** G

choose the longer sentences because we assume that they carry more relevant information.

2) Extract (subject, predicate, object)-triples via Stanford's information extraction framework, OpenIE, which takes the corpus as input (line 8). These triples, representing statements, are the basis of our second knowledge graph after removing duplicates.
3) Build the graph based on the first n_a statements (see Algorithm 2). 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 $G_s = (V_s, E_s)$ 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 (lines 12-16). Figure 4 illustrates this procedure for a sample sentence. The MATCH function connects two nodes if they share at least one token matches the other token.

IV. Argument Identification

The goals in this section are: 1) estimate the quality of our knowledge graph and confirm the contribution of our methods, 2) determine if the knowledge graph is beneficial to argument mining. The emerging field of argument mining aims to gather data with an argumentative context regarding a topic of interest to support decision-making. Following Stab *et al.* [27], we define an *argument* as a combination of a topic and a sentence holding evidence towards this topic.

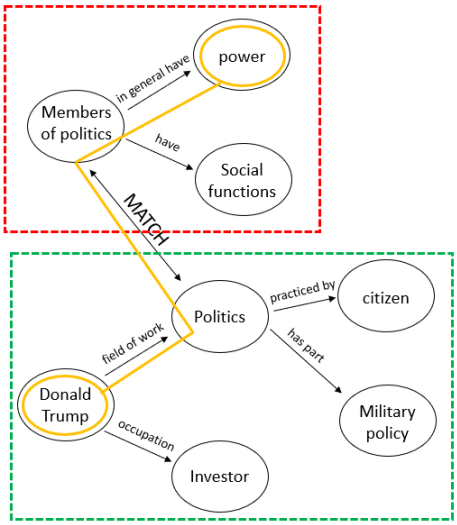

Fig. 4. Given the input sentence, "Donald Trump has a lot of power.", the structured data-driven knowledge graph (green box) and the unstructured data from Google (red box) are combined via a "MATCH" function. One can discover the yellow path, which would not have been possible with only one source of knowledge.

TABLE I
Argument Sentences:Cloning

Sentence	*Argument*
The same would be said of clones.	No Argument
DNA cloning has been used in genetic engineering to create plants that offer better nutritional value.	PRO
Cloning would undermine individuality and identity.	CON

The Argument dataset employed is from the UKP Sentential Argument Mining Corpus [28], and included eight topics. Our classifier predicts one of three classes: No Argument with respect to the topic, Argument for the topic (PRO), and Argument against the topic (CON). Table I presents examples of arguments on the topic of cloning.

A. Knowledge graph embedding

Before building the argument classifier using the KG paths, the KG triples are first represented as low-dimensional vectors. According to [29], KGs have three common patterns: (anti)symmetry, inversion, and composition. The KGs from the first part are antisymmetry patterns that are structured as $r(x, y) \implies \neg r(y, x)$ as given in the following example: (classification system, part of, library science), (class, part of, classification system). The capacities of TransE [30], ComplIEx [31] and RotatE [29] are usually used to get embeddings for those patterns. We selected TransE for the embedding in the following experiment. TransE is a simple and powerful transnational distance model [32]. Its goal is to complete the facts of a KG without additional knowledge. Its score function

is as follows:

$$- \parallel h + r - t \parallel_{1or2}, \quad (1)$$

where h and t refer to the entity's vector, and r is the relation connecting the h and t. For a triple, t would be the shortest distance between h and r. The distance can be measured as a L_1 or a L_2 function. However, one drawback of TransE is that it is not good at handling 1-to-N, N-to-N, or N-to-1 relations [32].

B. Experiment settings

Overall, there are two steps to the prediction: the first step is the sentence and knowledge graph encoder, and the second step is to transfer the vectors into a machine-learning model and predict the output. We compare three kinds of models: Models that only use knowledge graph embeddings but no sentence embeddings, BERT as a robust model that uses only sentence embeddings and no knowledge graph, and models that use both sentence embeddings and embedding representations for knowledge graph paths. Our model is an instance of the last category.

TransE + KG paths: In this model, the KG paths are the inputs. Each sentence in the KG model includes two types of paths: paths between entities and paths from the entities to the topic. We concatenate all sentences' paths in a topic and split the paths into triples (entity_1, relation, entity_2).

We use AmpliGraph, an open-source library for KG embeddings, to train all entities and relations within a topic by TransE. For each KG path, we get unique evidence embeddings and concatenate whole embeddings as a feature vector. Inspired by their classification example[4], the embedding features are the inputs to gradient boosted decision trees as implemented in Scikit-learn[5]. The max depth of the gradient boosted decision trees is three and the learning rate is set to 0.1 as default parameters.

BERT-KG: The inputs to our BERT-KG model are the sentences from the UKP Sentential Argument Mining Corpus and the KG paths for each sentence. The idea is similar to the baseline [6], where the authors concatenate the KG evidence to the XLNet pre-train model. We combined the two kinds of paths, paths between entities and paths between topics, in three models:

- sentence + entities,
- sentence + topic,
- sentence + topic + entities.

For most 'Non Argument' sentences, there are no extracted paths to the topic. The first step is KG path preprocessing: the paths for each sentence remove the duplicated relations and entities to reduce the length. With the help of the *Huggingface* 'BERT-large-case' pre-trained model, we get the sentence and the KG paths representation and transfer the feature vector as a new feature into the classifier. We utilize the pooler output from BERT pre-training as input into a linear regression layer with a softmax activation function in the experiments. Cross entropy is used as a loss function. All of the weights and biases are initialized randomly. The learning rate is set to $5e-6$, with early stopping if the accuracy on the development data did not improve after three epochs. The systems were run ten times with different seeds.

BERT-Sent: The inputs of the BERT-Sent model are the sentences without topic information and without KG information. The model of BERT-Sent is the same as BERT-KG, where both used the same BERT-large-case' pre-trained model, the learning rate and early stopping.

K-BERT: We chose the K-BERT model[6], introduced in the related work section, as the state-of-the-art model in our experiments. K-BERT brings a new perspective that is different from KG embedding, as it represents the entities in continuous vector spaces.

The authors create a lookup table with the key (entity_1), and as values all tuples (relation, entity_2) which match entity_1. In their original experiments, they used Chinese language datasets pre-trained on WikiZh. However, for our tasks, we instead use a pre-trained model on an English corpus: Google 'BERT-Base' model[7] with 12 hidden layers and a hidden dimension size of 768.

C. Experiments

Tabel II lists examples of triples on cloning, where the pieces of evidence are extracted using the LDA algorithm. The number of unique entities and relations of each knowledge graph is shown in Table III. The accuracy evaluation for all

TABLE II
EXAMPLE ENTITIES AND RELATIONS EXTRACTED BY USING LDA

Entity 1	*(predicted relation, Entity 2)*
Energy	(*part of*, universe), (*subclass of*, physical quantity)
Mitochondrion	(*has part*, Mitochondrial DNA), (*part of*, Cytoplasm), (*instance of*, cellular component)
Natural selection	(*subclass of*, biological selection), (*discoverer or inventor*, Charles Darwin), (*part of*, evolutionary biology)

TABLE III
UNIQUE NUMBER OF ENTITIES AND RELATIONS IN EACH TOPIC AND THE TRAIN/DEVELOPMENT/TEST SENTENCES

	Entities	*Relations*	*train*	*val*	*test*
Abortion	9435	186	2736	392	776
Cloning	5556	165	2083	297	635
Marijuana legalization	5356	133	1762	209	488
School uniform	6203	189	2135	285	585
Gun control	7321	171	2391	320	659
Nuclear	6261	144	2469	354	742
Minimum wage	4846	163	1679	267	462
Death penalty	5562	160	2492	376	718

[4]AmpliGraph https://docs.ampligraph.org/en/1.1.0/tutorials/ClusteringAndClassificationWithEmbeddings.html.

[5]https://scikit-learn.org/stable/modules/ensemble.html#gradient-tree-boosting.

[6]K-BERT: https://github.com/autoliuweijie/K-BERT.

[7]https://github.com/google-research/bert.

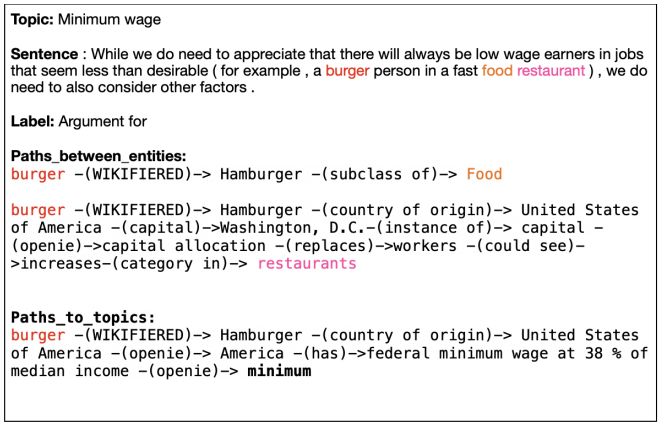

Fig. 5. A sample argument input (top) and its noisy concept paths between entities: burger to food and burger to restaurant. 'Paths between entities' and 'Paths to topics' are the outputs from the knowledge graph part.

experiments is given in Table IV. We computed each topic's macro averaged F1 score as the fraction between per-class F1 scores and the total number of classes.

TransE + KG paths: The results of KG extraction confirm that our retrieval methods can improve knowledge discovery for a small dataset. However, the results in Table IV indicate that knowledge graph evidence alone is not sufficient to outperform other methods.

K-BERT: Comparing with other state-of-the-art methods (e.g., K-BERT [23]), one drawback of our model is that the size of the knowledge graph is small. There is a lack of enough evidence to enrich the sentence. Also, the paths between entities are noisy. An example is shown in Figure 5: Here, the paths between entities are noisy, which reduces the accuracy of the argument classifier. For example, one path connects the concepts *burger* and *food* or *restaurants* within the sentence, even though they do not fit into the sentence-specific context.

BERT-KG: The three BERT-KG models use the same 'BERT-large-case' pre-trained model and parameters. It may come as no surprise that the BERT-based pre-trained models achieve better results overall in our comparison. As Table IV shows, BERT with sentences as input performs better on average than the vanilla BERT with KG information and the topic information. For the topics 'minimum wage' and 'death penalty', 'Sent+entities+topics' achieved the best F1 score among our own methods although the difference is not statistically significant. One shortcoming of our approach is that it cannot distinguish between positive and negative KG paths, i.e. it cannot decide whether the path supports the topic or not. An error example is shown in Table V. Overall the performance of our BERT-KG is comparable to that of BERT using only sentences as input. This indicates that the ability of our method to utilize the knowledge graph information needs to be improved.

V. Conclusion

We overcome 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. Compared to existing methods, we can process a considerably larger amount of data and consider more possible properties. This allows us to discover more relevant paths. This study confirms that KG enriches text understanding, and it is an essential focus for further research. In the future, we will continue improving our KG retrieval methods in the low-resource domain. One possible application is in neural topic models, which could be enriched with knowledge graph information. Further attention to KG embedding methods and the combination with sentence embeddings is also needed.

References

[1] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *Journal of Machine Learning Research*, vol. 3, p. 993–1022, 2003.

[2] T. Botschen, D. Sorokin, and I. Gurevych, "Frame- and entity-based knowledge for common-sense argumentative reasoning," in *Proceedings of the 5th Workshop on Argument Mining*. Association for Computational Linguistics, 2018, pp. 90–96.

[3] M. Fromm, E. Faerman, and T. Seidl, "Tacam: Topic and context aware argument mining," in *IEEE/WIC/ACM International Conference on Web Intelligence*, 2019, pp. 99 –106.

[4] P. Potash, R. Bhattacharya, and A. 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)*, 2017, pp. 342–351.

[5] P. Clark, O. Etzioni, T. Khot, D. Khashabi, B. Mishra, K. Richardson, A. Sabharwal, C. Schoenick, O. Tafjord, N. Tandon *et al.*, "From 'f'to 'a'on the ny regents science exams: An overview of the aristo project," *AI Magazine*, vol. 41, no. 4, pp. 39–53, 2020.

[6] S. Lv, D. Guo, J. Xu, D. Tang, N. Duan, M. Gong, L. Shou, D. Jiang, G. Cao, and S. Hu, "Graph-based reasoning over heterogeneous external knowledge for commonsense question answering," in *The Thirty-Fourth AAAI Conference on Artificial Intelligence*, 2020, pp. 8449–8456.

[7] M. Ostendorff, P. Bourgonje, M. Berger, J. Moreno-Schneider, G. Rehm, and B. Gipp, "Enriching bert with knowledge graph embeddings for document classification," *ArXiv*, vol. abs/1909.08402, 2019.

[8] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *ArXiv*, vol. abs/1810.04805, 2018.

[9] R. Wang, D. Tang, N. Duan, Z. Wei, X. Huang, J. ji, G. Cao, D. Jiang, and M. Zhou, "K-adapter: Infusing knowledge into pre-trained models with adapters," *ArXiv*, vol. abs/2002.01808, 2020.

[10] W. Xiong, J. Du, W. Y. Wang, and V. Stoyanov, "Pretrained encyclopedia: Weakly supervised knowledge-pretrained language model," *ArXiv*, vol. abs/1912.09637, 2019.

[11] Z. Zhang, X. Han, Z. Liu, X. Jiang, M. Sun, and Q. Liu, "ERNIE: Enhanced language representation with informative entities," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019, pp. 1441–1451.

[12] M. E. Peters, M. Neumann, R. Logan, R. Schwartz, V. Joshi, S. Singh, and N. A. Smith, "Knowledge enhanced contextual word representations," in *EMNLP-IJCNLP*, 2019, pp. 43–54.

[13] J. Guan, F. Huang, Z. Zhao, X. Zhu, and M. Huang, "A knowledge-enhanced pretraining model for commonsense story generation," *Transactions of the Association for Computational Linguistics*, vol. 8, pp. 93–108, 2020.

[14] B. He, D. Zhou, J. Xiao, X. jiang, Q. Liu, N. J. Yuan, and T. Xu, "Integrating graph contextualized knowledge into pre-trained language models," *ArXiv*, vol. abs/1912.00147, 2019.

TABLE IV

THE F1 MACRO SCORE FOR THE DIFFERENT ARGUMENT MINING TASKS. BERT-SENT, KBERT AND BERT KG MODELS WERE RUN TEN TIMES WITH DIFFERENT SEEDS. WE SET THE BERT-SENT AS THE BASELINE,'○' MEANS THERE IS NO SIGNIFICANT DIFFERENCE AT A SIGNIFICANCE LEVEL OF 0.05 BETWEEN BERT-SENT AND THE BERT-KG MODEL USING THE WILCOXON TEST, OTHERWISE IS '●'.

Topic	**Macro F1 scores**						
	Sentence-level BERT				**BERT-KG**		
	mtl+bilstm+dip2016 [27]	BERT-Sent	TransE	K-BERT	sent+topic	sent + entities	sent+entities+topic
Abortion	-	**0.721**	0.374	0.567	0.705○	0.70●	0.701●
Cloning	-	**0.750**	0.317	0.516	0.726●	0.737●	0.725●
Marijuana legalization	-	**0.705**	0.341	0.486	0.695○	0.699○	0.695○
School Uniform	-	**0.745**	0.302	0.482	0.729●	0.726●	0.731●
gun control	-	**0.657**	0.306	0.530	0.643○	0.640○	0.633●
nuclear	-	0.737	0.346	0.524	0.740○	0.734○	**0.742**○
minimum wage	-	0.716	0.418	0.544	0.708○	0.710○	**0.718**○
death penalty	-	**0.642**	0.319	0.434	0.615●	0.625○	0.628○
Average	0.4285	0.709	0.340	0.510	0.695	0.696	0.697

TABLE V

ERROR ANALYSES

Topic	Cloning
Sentence	Moreover , advocates of this objection caution against removing **God** from the process of creation altogether, which , it is argued , is what reproductive **cloning** achieves (Rikfin , 2000).
Golden Label	Argument Against
BERT-KG Predicted Label	Argument for
KG paths	cloning $->$ *form of is* $->$ In essence asexual method $->$ *openie* $->$ Scientific method, God $->$ *instance of* $->$ religious concept $->$ *subclass of* $->$ concept $->$ *said to be the same as* $->$ notion $->$ *justify* $->$ cloning laws
Error analyses	The extracted information does not consider sentiment information.

[15] A. Lauscher, I. Vulic, E. M. Ponti, A. Korhonen, and G. Glavas, "Informing unsupervised pretraining with external linguistic knowledge," *ArXiv*, vol. abs/1909.02339, 2019.

[16] Y. Levine, B. Lenz, O. Dagan, O. Ram, D. Padnos, O. Sharir, S. Shalev-Shwartz, A. Shashua, and Y. Shoham, "Sensebert: Driving some sense into bert," *ArXiv*, vol. abs/1908.05646, 2019.

[17] D. Paul, J. Opitz, M. Becker, J. Kobbe, G. Hirst, and A. Frank, "Argumentative relation classification with background knowledge," *Proc. COMMA, to appear*, 2020.

[18] J. Kobbe, J. Opitz, M. Becker, I. Hulpus, H. Stuckenschmidt, and A. Frank, "Exploiting background knowledge for argumentative relation classification," in *2nd Conference on Language, Data and Knowledge (LDK 2019)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2019.

[19] C. D. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. J. Bethard, and D. McClosky, "The Stanford CoreNLP natural language processing toolkit," in *Association for Computational Linguistics System Demonstrations*, 2014, pp. 55–60.

[20] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," in *NAACL-HLT, Volume 1 (Long Papers)*, 2018, pp. 2227–2237.

[21] A. Talmor, J. Herzig, N. Lourie, and J. Berant, "Commonsenseqa: A question answering challenge targeting commonsense knowledge," in *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 2019, pp. 4149–4158.

[22] L. Page, S. Brin, R. Motwani, and T. Winograd, "The pagerank citation ranking: Bringing order to the web." Stanford InfoLab, Technical Report 1999-66, November 1999.

[23] W. Liu, P. Zhou, Z. Zhao, Z. Wang, Q. Ju, H. Deng, and P. Wang, "K-bert: Enabling language representation with knowledge graph," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 03, 2020, pp. 2901–2908.

[24] A. Lauscher, O. Majewska, L. F. Ribeiro, I. Gurevych, N. Rozanov, and G. Glavaš, "Common sense or world knowledge? investigating adapter-based knowledge injection into pretrained transformers," *arXiv preprint arXiv:2005.11787*, 2020.

[25] J. Brank, G. Leban, and M. Grobelnik, "Annotating documents with relevant wikipedia concepts," in *SiKDD*, 2017.

[26] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *EMNLP*, 2014, pp. 1532–1543.

[27] C. Stab, T. Miller, B. Schiller, P. Rai, and I. Gurevych, "Cross-topic argument mining from heterogeneous sources," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2018, pp. 3664–3674.

[28] C. Stab, T. Miller, and I. Gurevych, "Cross-topic argument mining from heterogeneous sources using attention-based neural networks," *arXiv preprint arXiv:1802.05758*, 2018.

[29] Z. Sun, Z.-H. Deng, J.-Y. Nie, and J. Tang, "Rotate: Knowledge graph embedding by relational rotation in complex space," *arXiv preprint arXiv:1902.10197*, 2019.

[30] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko, "Translating embeddings for modeling multi-relational data," in *Neural Information Processing Systems (NIPS)*, 2013, pp. 1–9.

[31] T. Trouillon, J. Welbl, S. Riedel, É. Gaussier, and G. Bouchard, "Complex embeddings for simple link prediction," in *International Conference on Machine Learning*. PMLR, 2016, pp. 2071–2080.

[32] Q. Wang, Z. Mao, B. Wang, and L. Guo, "Knowledge graph embedding: A survey of approaches and applications," *IEEE Transactions on Knowledge and Data Engineering*, vol. 29, no. 12, pp. 2724–2743, 2017.