Text2Triple: Extracting Knowledge Graphs from Text Documents

Lixian Liu
University of Waterloo
Waterloo, Canada
l377liu@uwaterloo.ca

Ipsita Mohanty
University of Waterloo
Waterloo, Canada
imohanty@uwaterloo.ca

ABSTRACT

Knowledge Graph extraction from natural language either requires expensive supervised training, or relies heavily on manual input. We propose Text2Triple, a fully unsupervised system that extracts knowledge graphs in the form of relation triples from text documents. We leverage the dependency tree architecture of sentences to extract triples and map them to existing open knowledge base like Wikidata for standardization. We then evaluate our system's performance qualitatively with a set of handcrafted inputs. We also show that our system outperforms a state-of-the-art relation extraction system (Stanford OpenIE) on a gold standard metric. The code for Text2Triple is hosted on GitHub.

CCS CONCEPTS

• **Knowledge Graph Construction → Unsupervised model**; • **Natural language processing**;

KEYWORDS

knowledge graphs, natural language processing, open extraction, dependency tree

1 INTRODUCTION

Knowledge graphs (KG) provide a powerful representation of entities and the relationships between them. It is used to store and retrieve information in a manner that both computers and humans can comprehend and process. Knowledge graphs are often utilized in search engines, where they are used to deliver more relevant and tailored results for searches. For example, a search for "pizza" may give results containing information about various pizza varieties, local pizza restaurants, and related subjects such as Italian food. Knowledge graphs are also used in natural language interfaces to help computers comprehend and synthesize human language more efficiently. For example, a chatbot that employs a knowledge graph may be able to deliver more accurate and valuable replies to user inquiries as it has access to a detailed representation of real-world entities and their relationships. At present, various industries, including healthcare, finance, and e-commerce, are using knowledge graphs in applications involving semantic search, automated fraud detection, intelligent chatbots, advanced drug discovery, dynamic risk analysis, content-based recommendation engines, and knowledge management systems. For example, a knowledge graph might be utilized in healthcare to keep track of and integrate patient information, medical research, and treatment alternatives to promote tailored and evidence-based care.

Fall 2022, CS848, University of Waterloo
2022. ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

Unsupervised knowledge graph construction techniques enable systems to automatically learn and extract information from vast volumes of unstructured data without needing any manual assistance. This makes constantly changing data easier to integrate and also saves time and resources. It also guarantees the completeness of the graph and ensures its free from manual errors. This paper focuses on constructing knowledge graphs from plain text by extracting relation triples. We propose a system that extracts efficiently extracts relation triples from a text document and converts it into a Knowledge Graph with normalized relation triples. We borrow this idea from Stanford OpenIE [1], an unsupervised system to automatically extract structured information from unstructured text. The authors use linguistic analysis to determine the relationships between entities found in the text document and then generate a graph-based representation of the retrieved data. This results in more accurate and thorough information extraction, which is extremely useful in open-domain scenarios as it generally has complicated or implicit relations. To limit out-of-domain text and long-range dependencies in the extracted relations, it is important to have a fixed vocabulary for the Relation in {Subject, Object, Relation} triple. To address this, the system uses a manual procedure for mapping the extracted relations to a pre-defined relation schema. Further, the system uses a template-based approach to extract entities and their relations from a text. Although the authors limit the number of templates to a relatively small number to ensure in-domain relationships between entities, however, given a complicated or noisy unstructured input, it is difficult to cover all scenarios using a template-based method. Thus, given a random text document, achieving the quality of relation extraction as indicated in their study is challenging. Further, as it relies on a manual mapping procedure, it isn't easy to scale it up for large systems.

To improve this, we propose a Text2Triple, a system that doesn't use a template-based method for relation normalization and needs no manual mapping of extracted relations to the relation schema while making a knowledge graph. We show that our system produces equivalent results to the state-of-the-art unsupervised system, Stanford OpenIE. Our unsupervised approach is divided into two parts: one extracts a triple in the format of subject-relation-object in a sentence by leveraging the structure of the dependency tree of such a sentence, and the other maps the extracted relations to an open dataset and uses semantic similarity to map the relations to a pre-defined relation schema.

The major contributions made by this paper are as follows:

(1) We propose an unsupervised method for constructing knowledge graphs from text documents that produce results similar to the state-of-the-art system, Stanford OpenIE.
(2) We develop an entity extraction method that does not rely on providing pre-designed templates.

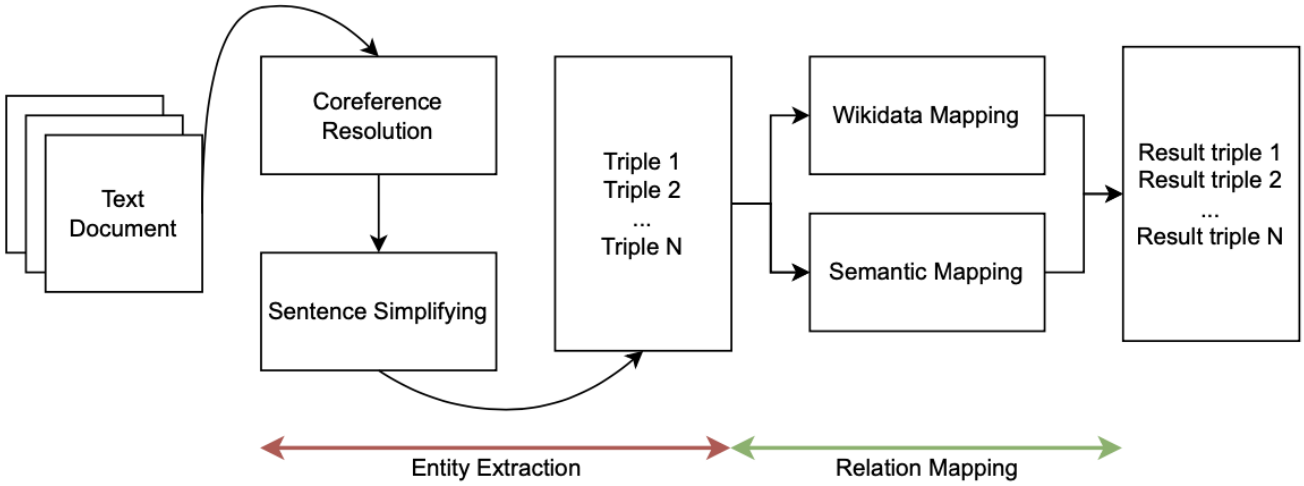

Figure 1: A system overview of the proposed methodology

(3) We design a technique that eliminates the need for manual mapping of extracted relations to a pre-defined relation schema.

The rest of the report is structured as follows. Section II provides a brief overview of previous works concerning this problem. Section III splits the task of KG extraction from text documents into three parts: 1) we first pre-process the text to segment it into concise clauses, 2) we then extract relation triples from these clauses by analyzing their grammatical structure 3) we finally standardize the extracted relations by mapping them to a large existing knowledge base (here Wikidata). We then evaluate our methods empirically via a set of experiments in Section IV. Finally, we conclude our works and point to future works.

2 RELATED WORK

Previous studies have shown the importance of knowledge graphs in various applications. In [3], the authors investigate how far knowledge representation and reasoning may advance by describing knowledge with graphs (in a graph theoretic sense) and reasoning it using graph operations. Various refinement strategies have been suggested to boost the value of such knowledge graphs, which attempt to infer and add missing knowledge to the graph or discover erroneous bits of information. [8] further presents an overview of such knowledge graph refining techniques, with a focus on both the methods suggested and the evaluation methodologies used.

In [12], the authors investigate an automated technique for learning high-quality knowledge bases by directly correlating illnesses and symptoms from electronic medical data. [16] introduces a unique open-domain conversation generation model to show how large-scale commonsense knowledge may aid in language comprehension and generation. [5] highlight current accomplishments and prospective research directions to aid future research. The survey [5] provides a comprehensive review of the knowledge graph covering overall research topics such as 1) knowledge graph representation learning; 2) knowledge acquisition and completion; 3) temporal knowledge graph; and 4) knowledge-aware applications, as well as summarize recent breakthroughs and future research directions, and this survey offers a comprehensive classification as well as new taxonomies for these themes.

In the past decade, owing to the increasing popularity of knowledge graphs and their potential use in various domains, numerous approaches have been employed for generating knowledge graphs from diverse sources. [10] utilizes partitioning algorithms that use ontological and distributional information to construct knowledge graphs from noisy extractions obtained from the web automatically. [14] summarises the existing techniques for constructing knowledge graphs developed by academia and the industry. They extensively discuss about the process of building knowledge graphs and survey state-of-the-art techniques for automatic knowledge graph checking and expansion via logical inferring and reasoning. In [7], the authors introduce a multi-task configuration for finding entities, relations, and coreference clusters in scientific papers. The multi-task setup reduces cascading errors between information extraction tasks and leverages cross-sentence relations through coreference links. [13] presents a novel framework for constructing knowledge graphs from open online encyclopedias. The system extracts the contents of targeted articles periodically and performs differential updates on the constructed KG. They utilize heuristic-based entity-matching strategies and a semi-supervised learning method to find duplicated entities and properties from various online encyclopedias.

The Web API community has amassed a plethora of information that may be utilized to improve KG construction in recent years; however, the current problem of having a fully automated, unsupervised system that effectively extracts KGs from unstructured data/text persists. Despite the active contributions from the scientific community to design engines to solve the challenge of knowledge graph generation, a lack of testbeds have precluded reliable benchmarking of these engines. To address this problem, [2] investigates and experimentally tests a collection of factors and

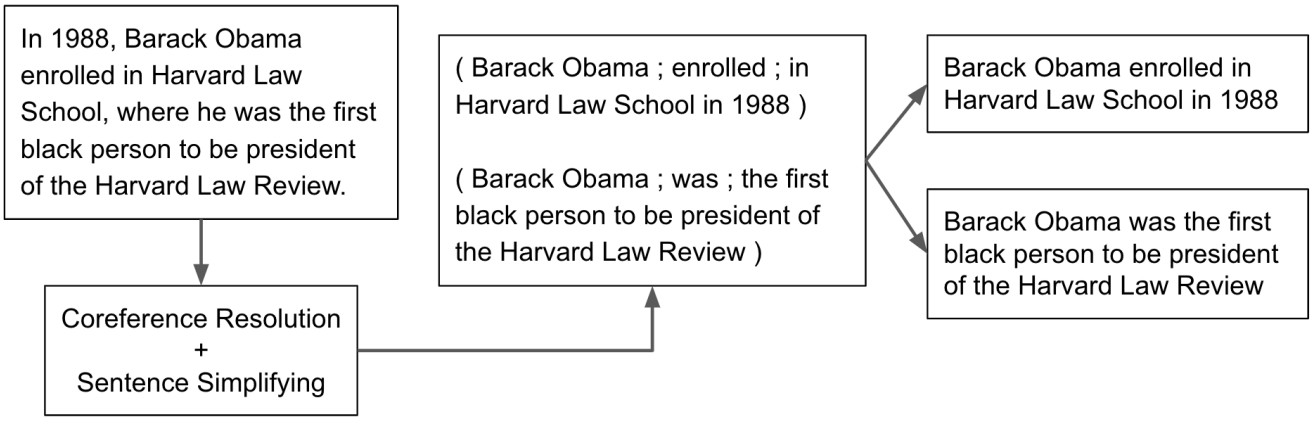

Figure 2: An illustration of the output of IMoJIE given a complicated sentence

settings that influence the behavior of such relation-extraction engines (e.g., data size, data distribution, mapping complexity). In [15], the authors propose a flexible approach for knowledge graph construction with minimal supervision based on unstructured biomedical domain-specific contexts, which includes entity recognition, unsupervised entity and relation embedding, latent relation generation via clustering, relation refinement, and relation assignment to assign cluster-level labels. [9] offer a semi-supervised entity alignment approach (SEA) that uses both labeled and unlabeled entity information for alignment. A recent unsupervised technique [1] developed by Stanford's NLP team extracts entities, and the relationships between them from text documents have shown promising results. It leverages the linguistic features of a sentence for entity-relationship extraction and uses a template-based approach to map the extracted relations to an existing schema.

3 METHODOLOGY

This section discusses our approach for extracting a set of entity-relation triples from a text document. Our approach is divided into two key components: entity extraction and relation mapping. Figure 1 illustrates the overall system overview, which we will discuss thoroughly in the following sections.

3.1 Sentence Segmentation

Given a text document containing numerous sentences, we aim to extract triples in the (Subject, Relation, Object) pattern, with the verb in each sentence indicating the relationship. When we work with natural language text, we frequently deal with compound (sentences with two parts that are equally important) and complex sentences (sentences with one part depending on another). Thus, it becomes sometimes useful to split these composite sentences into their component clauses for easier processing down the line. Many traditional systems use regular expression-based methods for sentence segmentation. However, regular expression-based systems are fragile and need frequent manual updates, and are often insufficient to segment a sentence into independent clauses. An independent clause is a group of words that contains a subject and a predicate and forms a complete thought when standing alone. It should ideally only have one subject and one predicate. For example, the sentence *He eats cheese, but won't eat ice cream* has two predicates. It should ideally be split into (*he eats cheese*) and (*he won't eat ice cream*). Thus, such complex sentences need to be segmented into clauses to avoid missing diverse and significant information while constructing the KG.

We use an Open Information Extraction (OpenIE) model named IMoJIE [6] to segment sentences into independent clauses. It is a highly efficient model with a very high accuracy rate in extracting structured information from natural language text. They use a generative approach to get a variable number of diverse extractions per sentence. The resulting clauses always have the original sentence's subject as the first entity, the verb as the relationship, and the remaining words in as the predicate.

While IMoJIE is often seen to generate long word phrases for the object, it is great at breaking down a complex sentence into sub-sentences having just a predicative verb, which is important for the subsequent work of entity extraction and relation mapping. To make the sub-tasks in our methodology smoother, we rebuild a phrase using IMoJIE's output file, as illustrated in Figure 2.

In addition to IMoJIE's capacity to segment a sentence into independent clauses, it also addresses the issue of conference resolution. Coreference resolution is the task of finding all expressions that refer to the same entity in a text and the primary goal of having a coreference resolution model is to detect distinct textual expressions of the same entity. For example, for the sentence *"Abraham Lincoln was born in Larue County, and he was the president of the United States"*, *"he"* stands for *"Abraham Lincoln,"* and the coreference resolution model should replace *"he"* with *"Abraham Lincoln"* in all extracted clauses. They use AllenNLP [4] to handle this issue by detecting coreference clusters. Figure 2 illustrates the output of IMoJIE for a sample sentence.

3.2 Entity Extraction using Dependency Tree

After extracting a set of simple and independent clauses from complex sentences using IMoJIE, we need to extract triples from these

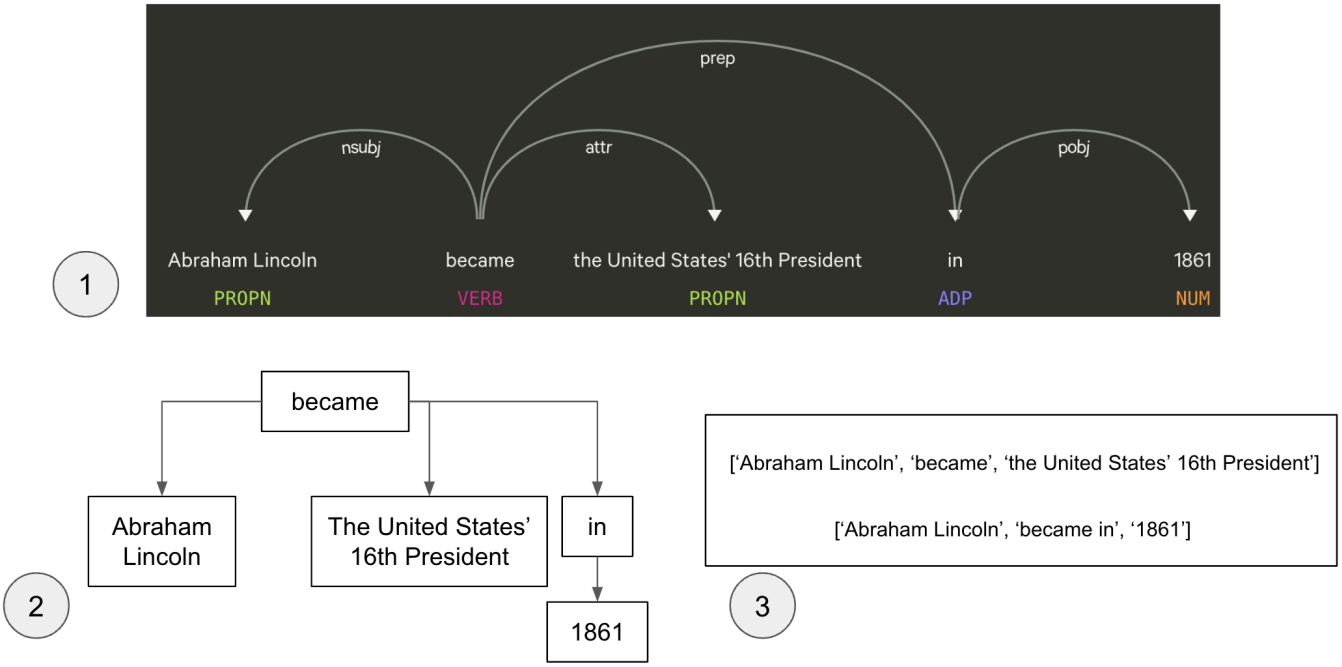

Figure 3: An example of the triple extraction algorithm using the dependency tree. Part 1) A dependency tree generated by spaCy given a phrase, Part 2) An output of the tree structure based on the input dependency tree, and Part 3) Sample triples extracted from the tree-based structure.

clauses. We use the dependency tree structure of these clauses to extract relation triples. A dependency tree is a grammatical structure added to a sentence or phrase which delineates the dependency between a word (such as a verb) and the phrases it builds upon (such as the subject and object phrases of that verb). We use **spaCy**, an open-source Python library for advanced natural language processing, to get the dependency tree structure of these short phrases. In a sentence's dependency tree structure, the tree's root is the main verb of the sentence. We mark this root as the relationship between a subject and an object, i.e., the root forms the Relation in the (Subject, Relation, Object) triple. Further, as IMoJIE's output always outputs clauses with the first entity as the subject of the sentence, we proceed with the assumption that the words in the left subtree of the root will always be the Subject. We iterate through each node in the right subtree of the root and return a word phrase as the object if the node in consideration is a noun, proper noun, or named entity recognition (NER). We do this until we reach the leaf of all the subtrees. If the node under consideration is a verb, we consider it as a root and add it to a queue. After processing the current root, we then process all new-found roots in the queue one by one by considering their corresponding subtrees. This approach is illustrated in Figure 3.

However, the above-mentioned algorithm may generate certain trivial triples, For example, for the phrase *"Abraham Lincoln became the United State's 16th President in 1861"*, generates a triple *['Abraham Lincoln', 'became in', '1861']* as shown in Figure 3.

Thus, if we add specific attributes to the relationship, we may generate fewer unnecessary triples and make each triple more informative. That is, if the object in a triple contains a location or a time component, we remove that from the object string and add it as an attribute of the triple. We are essentially creating an extended triple with Location and/or Time attributes for the phrases which contain this type of information. We modify the original Object structure to prevent redundancy and avoid unnecessary edges. Since multiple events may have occurred at the same time or location, this method of encoding relation triples might prove helpful for catering to the needs of QnA systems powered by KGs. Thus the final extracted triple from the phrase in Figure 3 now becomes [*'Abraham Lincoln', 'became', 'the United States' 16th President', 'Time: 1861'*].

3.3 Normalizing Relation in extracted triples

In the open domain, we encounter an extensive vocabulary. When we normalize text, we attempt to reduce its randomness, bringing it closer to a predefined "standard". This helps us to reduce the amount of different information that the computer has to deal with, and therefore improves efficiency. For our problem of KG construction, this is a highly crucial task in order to have standard relations across all extracted triples. For example, we may generate the triple *['Abraham Lincoln', 'be', 'president']*, but the Relation in this triple is neither standardized nor instructive. To solve this problem of normalization, we map the extracted relations to a standard vocabulary. We use two approaches to do so, which are discussed below.

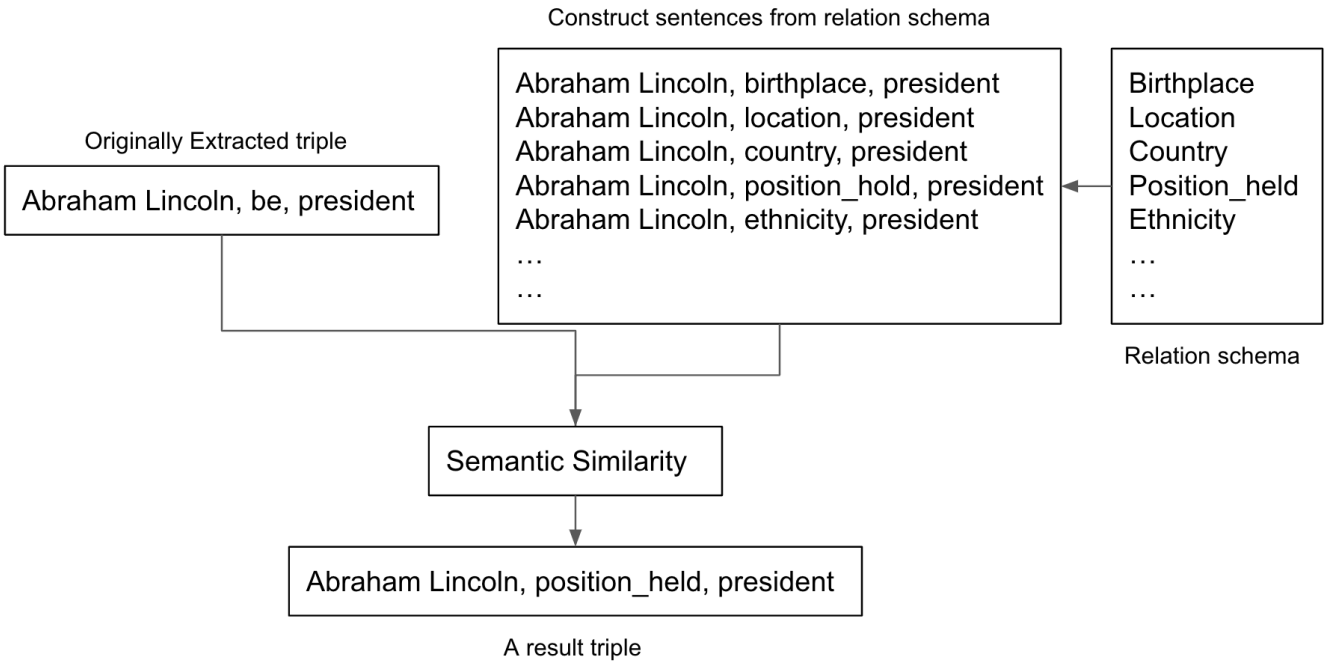

Figure 4: A procedure of using semantic similarity to match the relation from a pre-defined relation schema to the extracted triple

Relation Mapping with Wikidata. First, we try to map the extracted relations to WikiData, an open KG with over 100 million items and 8,000 distinct relation types. For an extracted triple (Subject, Relation, Object), we first get the unique identifier or WikiID of the Subject from WikiData using SPARQL queries and use this WikiID to fetch all its properties from WikiData. The properties are fetched from WikiData are in the form of [PropertyName, PropertyValue] tuples. We then compute a similarity score between the Object and fetched PropertyValues using the longest contiguous subsequence matching technique. This algorithm is implemented by using an open-source Python library called diffLib. If this similarity score is more than a specific threshold (0.8 in our case), then the Relation component in the triple is replaced by the corresponding PropertyName. If no PropertyValue matches with the extracted Object, the original relation in the triple is preserved, and we use the next described technique for relation normalization.

Relation Mapping with Semantic Similarity. The above approach ensures that the vocabulary of Relation is limited to what we see in WikiData. However, there are two main issues associated with this approach. First, there may be no record available in WikiData corresponding to the Subject or Object, in which case our SPARQL query fetches no result. Second, a user may want to fix the Relation vocabulary to a desired domain that she specifies. In both circumstances, simply mapping the extracted relations to WikiData is inadequate.

Thus, we offer a semantic similarity-matching approach to address these issues. Given a list of triples with Subject or Object that either do not match with any of the Wikidata entities or Relations that need to map to a pre-defined relation schema, we do the following. First, we construct a sentence using (Subject, Relation, Object) triple and another set of sentences by using the Subject, Object, and relations from the predefined schema. If no such vocabulary or schema is specified, we consider the set of PropertyNames from WikiData as our default schema. Given two phrases, we then use Sentence-BERT [11] to convert them to vector representations and compute the semantic similarity of these two sentences using cosine similarity. The schema with the highest confidence value will then replace the original relation in the triple. Figure 4 effectively summarizes this method.

We now have completed the KG extraction task from a natural language text document. We present the evaluation our system in the following section.

4 EVALUATION

We empirically evaluate our approach in the context of a real-world end-to-end extraction of a knowledge graph from a text document task. First, we present a qualitative analysis of the relation triples extracted by our system from a simple text document. Then for comparing our output with Stanford OpenIE's output, we come up with a gold standard input for entity recognition and relation extraction. We use a set of random sentences from the New York Times journal as input to both the system and compare their outputs with a set of benchmark triples. We talk about both of these metrics in detail in the following subsections.

Input: **The USA is a North American country that consists of 50 states and 14 territories.**

Output:

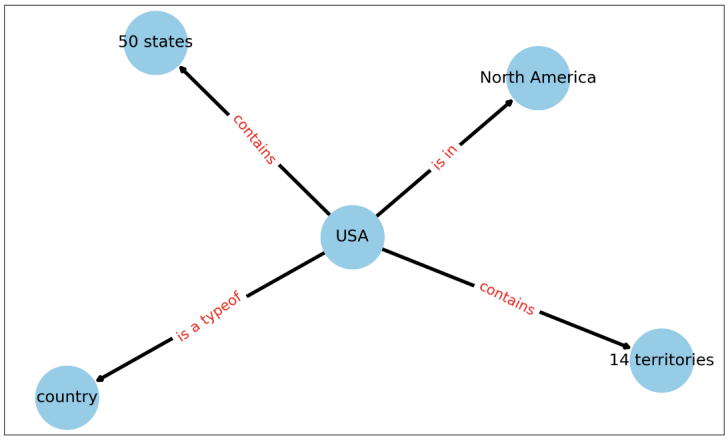

Figure 5: Graphical representation of KG extraction

4.1 Qualitative analysis

We take four text files with some random sentences and evaluate the precision and recall values achieved by our system. Each file contains 20-30 random facts specific to a randomly chosen country. We use this dataset because it is easy to verify the factual correctness of the extracted triples. The evaluation metrics are as follows -

- *Precision:* It measures the fraction of correct triples extracted among the total triples extracted by the system.
- *Recall:* It measures the fraction of relevant triples that were correctly extracted by the system

To get the benchmark relation triples, we refer to DBPedia and extract the triple corresponding to the listed fact in the input document. For example, for constructing the benchmark, the sentence *"The USA shares land borders with Canada and with Mexico* is manually mapped to relation triples *(USA, shares border with, Canada)* and *(USA, shares border with, Mexico)*. We do these mappings manually as existing systems fail to capture complex relationships between entities and thus are unsuitable for evaluation purposes.

Table 1 summarizes our results. Notice that in dataset number three, the system extracts more triples than we intuitively generate from the document. This is due to the unsupervised nature of our system, which extracts relationships between entities that humans often miss. For example, for the input sentence "The USA is a North American country," we extract triple *(USA, is in, North America)*. At the same time, our system outputs two triples - *(USA, is in, North America)* and *(USA, is of type, country)*. This shows our method can extract nominal relations, which is a direct improvement over most existing systems. Figure 5 illustrates the graphical representation of the extracted KG from a sample sentence.

4.2 Comparison with Stanford OpenIE

For comparing our system's output with OpenIE's performance, we consider a set of random sentences from the New York Times as the gold standard. (Subject, Relation, Object) triples have been extracted from these sentences manually via crowdsourcing. Out of the 50 sentences considered for comparison, it is seen that Text2Triple's output matches the correct triple for 28 sentences while Stanford's OpenIE's output only matches with 21 of the gold standard results. This is because when the input sentences are overly complex, it becomes important to segment the sentences into clauses and perform conference resolution on the extracted phrases. One of the sample sentences and the output for both systems is illustrated in Figure 6. Notice that although Text2Triple extracts some redundant triples, it captures some of the crucial properties of the entities (e.g., the triple {SUBJECT: Chandrika Kumaratunga RELATION: is a list of OBJECT: president}), and the extracted relations are consistent with the normalized vocabulary (i.e. "is" is changed to "is a list of")

5 CONCLUSION AND FUTURE WORK

We have presented a system for extracting open domain relation triples that break a long sentence into short, coherent phrases and then extract the maximally simple relation triples from it. For standardizing the extracted relations, we map the triples to an open knowledge base (WikiData) and use semantic similarity to normalize the Relation value. We evaluate our approach qualitatively with a handcrafted benchmark test and compare its performance with a state-of-art-system (Stanford's OpenIE) for relation extraction against a gold-standard corpus of data.

However, there are a few limitations to our method. Firstly, it relies on various third-party components at different steps of KG

Dataset No.	Total triples	Extracted Triples	Relevant Triples Extracted	Precision	Recall
1	29	34	25	0.74	0.86
2	38	52	34	0.65	0.89
3	21	32	26	0.81	1.0
4	30	29	26	0.89	0.86
Total	108	147	111	0.75	0.87

Table 1: Qualitative evaluation of our system on a randomly chosen corpus

Sentence	Stanford OpenIE output	Text2Triple output
Eric Fernando , a spokesman for Sri Lanka 's president , Chandrika Kumaratunga , said that aid had been abundant , but that there was a pressing need for water-purification equipment .	SUBJECT: Sri Lanka, RELATION: for, OBJECT: president SUBJECT: Eric Fernando, RELATION: spokesman for, OBJECT: Sri Lanka 's president SUBJECT: Sri Lanka, RELATION: 's president is, OBJECT: Chandrika Kumaratunga	SUBJECT: Eric Fernando RELATION: said to be the same as OBJECT: that aid had been abundant, but that there was a pressing need for water-purification equipment SUBJECT: Eric Fernando RELATION: is a list of OBJECT: a spokesman SUBJECT: Eric Fernando RELATION: is a list of OBJECT: spokesman for Sri Lanka's president SUBJECT: Chandrika Kumaratunga RELATION: is a list of OBJECT: president SUBJECT: Chandrika Kumaratunga RELATION: is a list of OBJECT: Sri Lanka's President SUBJECT: Chandrika Kumaratunga RELATION: is president of OBJECT: Sri Lanka

Figure 6: Comparison of Stanford OpenIE with Text2Triple

extraction. It would be interesting to create in-house methods for sentence segmentation and entity extraction. Further, the system can be extended to support other languages along with English to cater to every use case. After KG extraction, we can also have some module that computes materialization and extends the KG for datalog inputs. We could also work to come up with a better evaluation metric, as designing manual benchmarking metrics are time-consuming and often error-prone.

REFERENCES

[1] Gabor Angeli, Melvin Jose Johnson Premkumar, and Christopher D Manning. 2015. 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)*. 344–354.

[2] David Chaves-Fraga, Kemele M Endris, Enrique Iglesias, Oscar Corcho, and Maria-Esther Vidal. 2019. What are the parameters that affect the construction of a knowledge graph?. In *OTM Confederated International Conferences" On the Move to Meaningful Internet Systems"*. Springer, 695–713.

[3] Michel Chein and Marie-Laure Mugnier. 2008. *Graph-based knowledge representation: computational foundations of conceptual graphs*. Springer Science & Business Media.

[4] Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson Liu, Matthew Peters, Michael Schmitz, and Luke Zettlemoyer. 2018. Allennlp: A deep semantic natural language processing platform. *arXiv preprint arXiv:1803.07640* (2018).

[5] Shaoxiong Ji, Shirui Pan, Erik Cambria, Pekka Marttinen, and S Yu Philip. 2021. A survey on knowledge graphs: Representation, acquisition, and applications. *IEEE Transactions on Neural Networks and Learning Systems* 33, 2 (2021), 494–514.

[6] Keshav Kolluru, Samarth Aggarwal, Vipul Rathore, Soumen Chakrabarti, et al. 2020. Imojie: Iterative memory-based joint open information extraction. *arXiv preprint arXiv:2005.08178* (2020).

[7] Yi Luan, Luheng He, Mari Ostendorf, and Hannaneh Hajishirzi. 2018. Multi-task identification of entities, relations, and coreference for scientific knowledge graph construction. *arXiv preprint arXiv:1808.09602* (2018).

[8] Heiko Paulheim. 2017. Knowledge graph refinement: A survey of approaches and evaluation methods. *Semantic web* 8, 3 (2017), 489–508.

[9] Shichao Pei, Lu Yu, Robert Hoehndorf, and Xiangliang Zhang. 2019. Semi-supervised entity alignment via knowledge graph embedding with awareness of degree difference. In *The World Wide Web Conference*. 3130–3136.

[10] Jay Pujara, Hui Miao, Lise Getoor, and William W Cohen. 2013. Ontology-aware partitioning for knowledge graph identification. In *Proceedings of the 2013 workshop on Automated knowledge base construction*. 19–24.

[11] Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084* (2019).

[12] Maya Rotmensch, Yoni Halpern, Abdulhakim Tlimat, Steven Horng, and David Sontag. 2017. Learning a health knowledge graph from electronic medical records. *Scientific reports* 7, 1 (2017), 1–11.

[13] Tianxing Wu, Haofen Wang, Cheng Li, Guilin Qi, Xing Niu, Meng Wang, Lin Li, and Chaomin Shi. 2020. Knowledge graph construction from multiple online encyclopedias. *World Wide Web* 23, 5 (2020), 2671–2698.

[14] Jihong Yan, Chengyu Wang, Wenliang Cheng, Ming Gao, and Aoying Zhou. 2018. A retrospective of knowledge graphs. *Frontiers of Computer Science* 12, 1 (2018), 55–74.

[15] Jianbo Yuan, Zhiwei Jin, Han Guo, Hongxia Jin, Xianchao Zhang, Tristram Smith, and Jiebo Luo. 2020. Constructing biomedical domain-specific knowledge graph with minimum supervision. *Knowledge and Information Systems* 62, 1 (2020), 317–336.

[16] Hao Zhou, Tom Young, Minlie Huang, Haizhou Zhao, Jingfang Xu, and Xiaoyan Zhu. 2018. Commonsense knowledge aware conversation generation with graph attention.. In *IJCAI*. 4623–4629.