NOVA IMS Information Management School

MDSAA

Master Degree Program in

Data Science and Advanced Analytics

Triplet extraction leveraging sentence transformers and dependency parsing

Stuart Gallina Ottersen

Dissertation

presented as partial requirement for obtaining the Master Degree Program in Data Science and Advanced Analytics

NOVA Information Management School
Instituto Superior de Estatística e Gestão de Informação
Universidade Nova de Lisbo

NOVA Information Management School
Instituto Superior de Estatística e Gestão de Informação
Universidade Nova de Lisboa

TRIPLET EXTRACTION LEVERAGING SENTENCE TRANSFORMERS AND DEPENDENCY PARSING

by

Stuart Gallina Ottersen

Dissertation presented as partial requirement for obtaining the Master's degree in Advanced Analytics, with a Specialization in Data Science

Co Supervisor: Fernando Bação

Co Supervisor: Flávio Pinheiro

November 2022

STATEMENT OF INTEGRITY

I hereby declare having conducted this academic work with integrity. I confirm that I have not used plagiarism or any form of undue use of information or falsification of results along the process leading to its elaboration. I further declare that I have fully acknowledge the Rules of Conduct and Code of Honor from the NOVA Information Management School.

Trondheim, 17/11/2022

DEDICATION (OPTIONAL)

ACKNOWLEDGEMENTS (OPTIONAL)

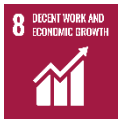
ABSTRACT

Knowledge Graphs are a tool to structure (entity, relation, entity) triples. One possible way to construct these knowledge graphs is by extracting triples from unstructured text. The aim when doing this is to maximise the number of useful triples while minimising the triples containing no or useless information. Most previous work in this field uses supervised learning techniques that can be expensive both computationally and in that they require labelled data. While the existing unsupervised methods often produce an excessive amount of triples with low value, base themselves on empirical rules when extracting triples or struggle with the order of the entities relative to the relation. To address these issues this paper suggests a new model: Unsupervised Dependency parsing Aided Semantic Triple Extraction (UDASTE) that leverages sentence structure and allows defining restrictive triple relation types to generate high-quality triples while removing the need for mapping extracted triples to relation schemas. This is done by leveraging pre-trained language models. UDASTE is compared with two baseline models on three datasets. UDASTE outperforms the baselines on all three datasets. Its limitations and possible further work are discussed in addition to the implementation of the model in a computational intelligence context.

KEYWORDS

Triple extraction; Natural language processing; Knowledge Graph

Sustainable Development Goals (SGD):



INDEX

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND ACRONYMS

NLP	Natural Language Processing
NER	Named Entity Recognition
KG	Knowledge Graph

1. INTRODUCTION

It is estimated that 85% of all data generated is unstructured (Boe, 2014).As such, information extraction and value generation require non traditional techniques that pose a challenge when facing the ever-growing volume of data produced and stored. In that context, text-based sources are particularly relevant source of information and potential value. The fields of Natural Language Processing and Text Mining have, over the years, introduced new techniques to quantify information from text-based sources and develop models to reproduce the nuances of human communication. In that context, Knowledge Graphs provide an interesting approach to store and model the knowledge structures of associations between entities and concepts. Although new, Knowledge Graphs have found applications in many domains such as query answering, recommender systems, search engines (Xiang, 2020) and fraud detection (Zhan, 2018).

Knowledge Graphs aim at representing knowledge structures through directed labelled graphs where nodes represent entities and links relationships between said entities. As such, Knowledge Graph map information into a structured set, T, of triplets. Each triplet, $t \in T: e_h, r, e_t$ with $e_h, e_t \in E$ and $r \in R$, considers e_h as to represent the head entity, e_t the tail entity, and r the relation between them. E is the set of entities and R is the set of relationships. However, contrary to traditional graph databases, Knowledge Graphs (Webber, 2021) assume that such structures obey a set of organising principles -- taxonomy or an ontology -- that can facilitate the insertion of new observations, the retrieval of meaningful knowledge, or the identification of missing (yet to be observed but likely to exist) relationships. Examples of popular Knowledge Graphs include Google Knowledge Graph (Singhall, 2022), DiffBot (diffbot, 2022), KBPedia (kbpedia, 2022).

Although Knowledge Graphs can be built case by case and through a careful insertion of relationships, a more traditional approach is to rely on automatic means to extract relationships between entities from text-base sources. Such an approach, is quite valuable in scenarios where the volume of information is large or when it is necessary to quickly access the 'essence' of a body of text. The challenge of building a knowledge graph can be simplified by breaking it down into its constituent tasks, these being entity extraction and relation extraction. Entity extraction is the process of identifying and parsing entities from text. These entities are then paired up and relation extraction is undertaken, where one tries to find the relation between the two entities from the context in the text. These two tasks together are one way of performing triplet extraction.

Knowledge graph construction methods can be split in two main approaches: unsupervised and supervised methods. Most current methods are supervised, but the requirement of labelled data to feed into such models is often expensive and time-consuming to create and not readily available (Singh, 2021), (Ulges, 2020), (Etzioni, 2012), (Etzioni A. F., 2011). Knowledge Graphs created with supervised algorithms are limited in the scope of entities they represent, as they restrict their domain to that of the labelled dataset. Moreover, supervised methods are also limited to a predefined set of relations (An, 2022).

Unsupervised methods overcome some of the limitations of supervised models. Examples, of unsupervised methods include OpenIE and KGSS. One of the most challenging problems these methods face is the proliferation of redundant triplets (An, 2022). In other words, the inability to disambiguate triplets that are, to all purposes, out of the scope of the Knowledge Graph being built. Additionally, the

process of triplet extraction is in some cases based on empirical rules, which, to a certain extent, defeats the purpose of automating the process. Here, we propose UDASTE, an unsupervised approach to triplet detection that aims and mitigate some of the shortcomings of current methods. It improves the task of triplet identification and extraction by leveraging the improved performance of named entity recognition, dependency parsing models, and MiniLM to extract relations. The availability of improved performance named entity recognition and dependency parsing models makes it enticing to leverage their capability for triplet extraction.

The rest of the document is organized as follows, Section 2 presents relevant theory and related work, Section 3 the UDASTE model, Section 4 the evaluation methodology, Section 5 the Results and Discussion, Section 6 a possible application of UDASTE and Section 7 concludes and presents limitations and further work.

2. RELATED WORK

In this section relevant models and theory will be discussed. Language representation models, dependency parsing and named entity recognition models are discussed as they are central to UDASTE. While in the following sections discuss models similar to UDASTE.

2.1. LANGUAGE REPRESENTATION MODELS

With the emergence of the Transformer architecture based on attention mechanisms, there have been considerable improvements over previous sequence transduction models (Vaswani, 2017). This transformer architecture has made significant improvements in language representation models such as in the Bidirectional Encoder Representation from Transformers, also known as BERT (Toutanova, 2018). BERT can be used in a wide array of tasks without considerable architectural change allowing more widespread use of language embeddings in research and applications with lacking resources to build custom solutions. Models such as BERT have then been made even more accessible through less computationally demanding, distilled versions, with similar performance in models such as MiniLM (Wang, 2020).

BERT is a language representation model that creates sentence embeddings by training deep bidirectional representations from unlabeled text (Toutanova, 2018). The model is bidirectional, meaning it is trained on text sequences both from right-to-left and left-to-right, which allows it to have a deeper 'sense' of language and understand the context words are used in. What MiniLM does to distil the teacher model, which in UDASTEs case is BERT, is to distil the self-attention module of the last layer of the teacher transformer model. This more lightweight model reduces the number of computations while maintaining high accuracy compared to the parent model. The presented model uses an implementation from the hugging face library based on the work in (Wang, 2020) that has been fine-tuned on 1 billion sentences[1].

2.2. DEPENDENCY PARSING

The dependency parser is used to extract grammatical dependencies between words, which aids in finding word pairs that can be fitting entity pairs for triplet extraction. Dependency parsers consist of a stack, buffer and a dependency graph and work their way through a sentence by moving words from the buffer to the stack. In each iteration one action can be selected from a set of actions, which will vary based on the type of algorithm used, through these actions arcs/dependencies are created between the words that are placed in the stack. One of the algorithms that exists is known as arc-eager with tree constraint, which allows postponing attachment decisions and ensures the creation of trees and not forests (Fernandex-Gonzalez, 2013). The dependency parser used in UDASTE, uses some of the concepts from the arc-eager model and combines it with a non-monotonic parsing model, which allows the model to correct prior misclassifications. This increases the set of complete parse trees, which allows the model to escape 'garden-paths' which is a common disambiguation problem (Johnson, 2015)[2].

[1] https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2

[2] Implementation of dependency parser: https://spacy.io/

2.3. Named Entity Recognition

Named entity recognition has a similar role to the dependency parser as it extracts pairs of words, which can be used to create entity pairs for triplet extraction. There are a wide variety of ways named entity recognition can be performed, most of the recent models with state-of-the-art results use deep learning. These deep learning models base themselves on architectures such as context encoder and tag encoder architectures (Li, 2022). The model chosen for UDASTE uses a less common transition-based architecture which identifies non-overlapping labelled spans of tokens. This architecture allows it to avoid the use of gazetteers/external information, which lends itself to better generalisability (Li, 2022). This transition model is conceptually similar to how the dependency parser works. This is implemented with the use of Stack-LSTMs and an additional transition that merges a series of words into chunks, which allows it to detect multi word entities (Singh N. S., 2022), (Dyer, 2016)[3].

2.4. Supervised models

Triplet extraction can be performed in a supervised or unsupervised fashion, where supervised models rely on labelled input. Models that fall in this category are RECON and SpERT. RECON uses a combination of graph neural network representations of sentences and facts from existing knowledge bases, this allows it to infer new triplets by combining triplets extracted from the unstructured text and preexisting facts taken from a knowledge base (Singh A. B., 2021). SpERT is an attention model for span-based joint entity and relation extraction. Where a sentence is tokenised, more infrequent words will be mapped to subwords, which allows the model to reduce the vocabulary size and map out-of-vocabulary words. Spans are created with each of the token subsequences, these spans are then categorised according to the type of entity they contain, they can also not contain any entity type. The spans containing entities are then paired together and SpERT estimates the relation, if any, based on a preexisting set of relations. The method requires training data to learn embedding sizes and relation classifiers and to fine-tune BERT (Ulges, 2020). There is also a category that falls between the two aforementioned methods, which is semi-supervised methods. Where we have ReVerb and OLLIE. Reverb defines syntactic and lexical constraints to identify relation phrases, that are then used to extract pairs of noun phrase arguments. This allows it to greatly reduce the number of incoherent extractions. These pairs are then assigned a confidence score using logistic regression, the confidence function is trained on labelled data from Wikipedia, which is what makes the model semi-supervised (Etzioni A. F., 2011). OLLIE uses ReVerb to get a set of seed tuples that it uses to learn open pattern templates, these templates are then applied to extract triplets. This allows it to address that only verb-mediated relations are extracted and context is ignored in ReVerb (Etzioni M. M., 2012).

2.5. Unsupervised models

Early unsupervised methods were mostly based on rule and heuristic methods such as YAGO and DBpedia. Both additionally extract a lot of their information from semi-structured sources such as Wikipedia (Suchanek, 2007), (Ives, 2007). More recently we have models such as OpenIE that generates knowledge graphs using binary relations produced by an Open Information Extraction approach combined with natural language processing. It does this by producing clauses that can

[3] Implementation of NER: https://spacy.io/

stand on their own both syntactically and semantically based on the original sentences. The generated clauses are then turned into maximally compact sentences that retain the core semantics of the original sentences. These compact sentences are then segmented into triplets with 6 simple dependency patterns. The clause generator and inference model can be trained, which could make the model be considered supervised, but in this paper, it is applied as an unsupervised model (Manning, 2015). OpenIE has a tendency to over-extract, creating too many unuseful relations, which can cause it to perform better recall-wise than precision (An, 2022). Another model worth mentioning is Match and Map (MAMA) which leverages existing language models such as (BERT, and GPT-2/3) without human supervision. It does this in two phases, the match phase where it uses beam search to find the best-matched candidate facts given a sentence. The following filter phase then reduces the number of facts by using a series of constraints (Toutanova, 2018), (Radford, 2019), (Song, 2020). Similarly, RE-Flex applies pre-trained language models to perform constrained cloze completion, which is predicting cloze/masked text. Given a target relation, an entity pair and a context it uses a cloze template that expresses the relation to create candidate context-entity pairs. These pairs go through candidate rejection where they are scored and the bottom portion of the distribution is rejected. Relation extraction is then performed on these pairs by identifying the subsequence in the corresponding context that completes the relation for the given entity (Rekatsina, 2020). DeepEx again uses pre-trained language models but uses a zero-shot approach that generates a series of candidate triplets based on attention scores and beam search. These candidates are then evaluated using a pre-trained model (Song C. W., 2021). Somewhat similarly in KGSS, a set of candidate relations are generated with a predefined set of relations and then these are scored using semantic similarity with the original sentence (An, 2022). To do this comparison SBERT is used (Gurevych, 2019). The model also leverages a set of empirical rules to filter and generate triplets, these might not always generalise well.

3. UDASTE MODEL

This section will discuss the model and its components. The model can be broken down into 6 components as seen in green/rounded boxes in Figure 3.1. The purple/larger rounded box is where the triplets are created. The two square boxes in blue represent the input data.

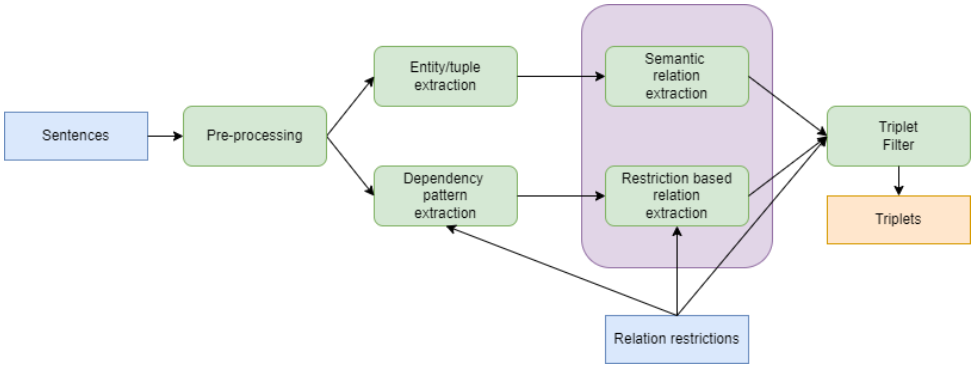


Figure 3.1 Flowchart showing structural components of UDASTE.

3.1. RELATION RESTRICTIONS

The relation restrictions are on the form $(mother)$: $[(PERSON, PERSON)]$ and are predefined, meaning that if a tuple of entities where both are of type $PERSON$, then the relation $mother$ is allowed. These 'restrictions' are later used to both create and then later on filter triplets. As part of these rules, it is also possible to set a 'reverse flag', which is specifically for symmetric entity type tuples. This is to handle both relations such as *contains* and *is_part_of*, which can both be part of the relation set, but the order of the entities would be reversed. For example in the case of *Italy, contains, Rome* and *Rome, is_part_of, Italy*.

3.2. PRE-PROCESSING

To better handle longer documents composed of multiple sentences some simple pre-processing is done before the triplet extraction begins. Coreference resolution is used to resolve references to previous entities, for example as seen in Figure 3.2. where 'he' is resolved to be referencing 'Barack Obama'. In the presented model the library NeuralCoref[4] is used to do this. Following this the document is split into sentences using sentence boundary detection.

[4] https://github.com/huggingface/neuralcoref

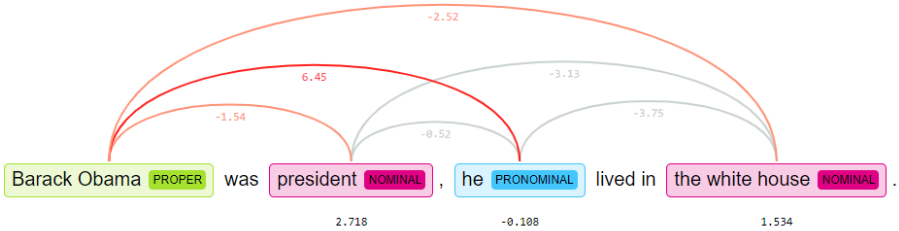


Figure 3.2 Chart showing coreference resolution.

3.3. Entity/tuple extraction

The entity extraction uses named entity recognition to extract the entities. The entities are kept in the order they appear in the sentences. Given an entity E_k at position k, it will create a pair with the entities E_{k+1} and E_{k+2}, meaning we get the pairs (E_k,E_{k+1}) and (E_k,E_{k+2}). This method was selected over using all permutations as pairs (E_k,E_{k+m}) with m>2 created issues with relation extraction. The relation extraction is performed with the text between the two entities, and with m>2, the span of text between the two entities contains multiple other entities, which can skew the semantic meaning compared to the predefined relations.

3.4. Semantic relation extraction

The relation extraction is done using MiniLM, the span between and including the entities, is compared to the MiniLM embedding of the predefined relations. These relations are taken from the predefined list. The comparison between the sentence span and the relations is done using cosine similarity. Only the most similar relation is selected when finally predicting a triplet.

Some of the entity pairs extracted will be adjacent, causing the above method to not work as there is no meaningful embedding to compare with. Adjacent entity pairs are therefore skipped. The section below discusses an additional method that attempts to leverage grammatical dependencies to handle some of these adjacent tuples.

3.4.1. Dependency pattern extraction / Restriction-based relation extraction

This component of the model leverages dependency parsing with the restrictions in 3.1. This is done by using apposition tuples which are defined as ''a grammatical construction in which two or more usually adjacent words, phrases, or clauses (especially nouns or noun equivalents) that have the same referent stand in the same syntactical relation to the rest of a sentence...''[5]. An example of an apposition would be "the president" in the phrase 'Sam, the president'.

[5] https://www.merriam-webster.com/dictionary/apposition

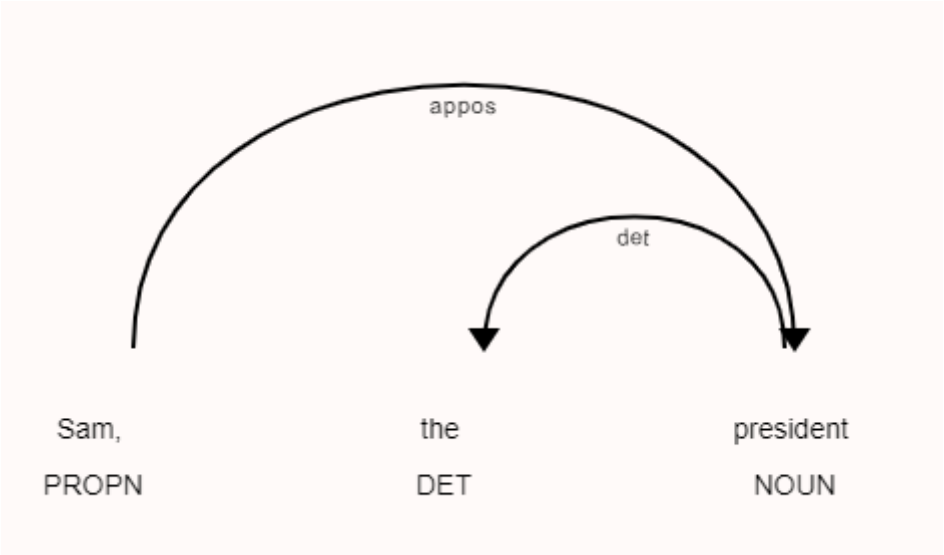


Figure 3.3 Dependency chart showing appositional dependency.

Using the aforementioned sentence we would get the entities *sam* and *president* as seen in Figure 3.3.This would correspond to the entity types *PERSON* and *PERSON*, using these entity types one can then find possible relations using the restrictions discussed in 3.1. Depending on the entity types this can result in a fairly large set of possible relations, which makes it crucial to define a set of relations that do not have many similar entity-type tuples.

The words extracted using appositions are run through a function that uses dependency parsing to extract spans of compounds. An example would be extracting 'DuVernay', the function would expand this to 'US filmmaker Ava DuVernay' using a compound dependency. This causes the multi-token entity problem to arise, where each token in the entity can have its own entity type. This has been solved by using the root of the multi-token entity and its entity type.

3.5. TRIPLET FILTER

The triplet filter is an essential part of limiting the number of erroneous triplets. It is based on using the restrictions discussed in 3.1. Each triplet contains two entities, and each of these has an entity type, this is used to find a corresponding list of 'allowed' relations, if the relation that appears with the pair of entities is not in the list the triplet is filtered out. The entity type is extracted as mentioned using the root of the span of the entity.

4. METHODOLOGY

4.1. DATASETS

To gauge the performance of the model the datasets used are WebNLG, NewsKG21 and NYT (An, 2022), (Perez-Beltrachini, 2017), (Zhao, 2018). Below in Table 4.1 some key statistics on the three datasets are available. The number of sentences and triplets is based on the test set of the given dataset.

Table 4.1 Table showing the datasets used, the number of sentences and the number of relations.

Dataset	Documents/Sentences	Triplets	Relations
NewsKG21	685	1247	91
WebNLG	14476	14476	211
NYT	5000	8616	24

4.1.1. NewsKG21

This dataset consists of sentences taken from news articles published in 2021 taken from CNN, CBC, USNEWS, The Star and Wikipedia News. The triplets were manually annotated (An, 2022).

4.1.2. New York Times

The dataset used from the New York Times is a dataset consisting of articles published between 1987 and 2007 taken from (Sandhaus, 2008). This dataset has then been labelled using distant supervision with Freebase as an external supervision source (McCallum, 2010), (Taylor, 2008). The dataset used in the testing here is an adaptation of this original dataset where all sentences of 100 words and above are removed (Zhao, 2018)[6].

4.1.3. WebNLG

This dataset differs from the others as it is a computer-generated dataset based on a knowledge base, and presents sentences that are more reminiscent of facts or statements rather than sentences one might read in day-to-day situations. The WebNLG dataset gets its name from the framework that extracted the dataset from an existing knowledge base. The version used in the testing is based on data from DBPedia (Perez-Beltrachini, 2017).

[6] Dataset used was found here: https://www.kaggle.com/datasets/daishinkan002/new-york-times-relation-extraction-dataset?resource=download

4.2. DATA PREPARATION

Each of the datasets requires a list of relations and the entity type tuples allowed for each of the relations. The list of relations was found with the dataset and if no predefined list of relations exists, the relations are extracted from the annotated triplets in the given dataset. The relations were then manually assigned entity-type tuples based on the meaning of the relation or the usage of the relation in the corresponding train dataset.

4.3. EVALUATION MEASURES

F1-score is used as the main evaluation method as it considers both recall and precision of the extracted triplets. Recall is a measure of the proportion of actual correct triplets that were identified. It can be calculated as the number of correctly identified triplets divided by the number of annotated triplets. Precision on the other hand aims to measure the proportion of the predicted triplets that were actually correct. It can be calculated as the number of correctly identified triplets divided by the total number of predicted triplets. To create a good triplet extraction model one wants to have a balance of the two. Meaning trying to extract as many of the correct triplets as possible while not predicting too many. The formula for F1 score is $F1 = \frac{2*precision*recall}{precision+recall}$.

4.3.1. Token set ratio

To compare the entities extracted Token set ratio is used, this evaluation technique was inspired by (An, 2022). Given an entity E and the true entity G, the token set ratio is calculated as $\frac{2M}{T}$ where T is the number of tokens in G and E combined, meaning |G|+|E|, M is the number of matching tokens in G and E. The threshold used in our evaluation is empirically set to 0.9, to make the values comparable with the results from (An, 2022).

4.4. BASELINE MODELS

As benchmarks to compare score the KGSS model from (An, 2022) and OpenIE from (Manning, 2015) will be used. UDASTE builds on the same idea as KGSS, of using embeddings to compare meaning, it is therefore considered a good benchmark. The KGSS implementation used was taken from the GitHub page provided in the paper (An, 2022)[7]. OpenIE leverages the dependencies within the sentences to extract clauses for triplet extraction, this is somewhat similar to what is done in UDASTE, making it an apt comparison. The two models were selected due to their similar running time, ease of access to implementations of the models and due to some interesting conceptual similarities.

To make the comparison fair, the relations extracted by OpenIE are mapped to the relation set in the dataset at hand using embeddings and cosine similarity. Although OpenIE can be considered a supervised model, in this evaluation it is considered unsupervised as no additional training is done.

[7] https://github.com/lixianliu12/KGSS

5. RESULTS AND DISCUSSION

As seen in Table 5.1 the two triplet extraction components complement each other well with no overlap of the correctly predicted triplets in NewsKG21 and limited overlap in NYT. The filtering is essential as seen in the number of extracted triplets, it removes about 83% and 75% of the predicted triplets, with only a 9% and 22% reduction in the correct triplets for NYT and NewsKG21 respectively. The relatively similar values before and after filtering are good, but also indicate that the model without the filter has a ceiling considering recall. UDASTE, even with a large number of extracted triplets, struggles with correctly predicting even 15% of the annotated triplets in NYT.

Interestingly the performance of the semantic extraction and rule-based extraction have drastically different proportions in the NYT dataset compared to NewsKG21. In the former the restriction based extraction provides around $\frac{2}{3}$ of the correct triplets, while in NewsKG21 only around $\frac{1}{3}$ .The datasets should be relatively similar as they both base themselves on news articles, one possible reason for the difference in performance is longer sentences, with the average length of the sentences being 37.7 words in NYT and 28.4 in NewsKG21, it could also come down to the writing style of the sources of the news articles or the way the triplets are annotated.

Table 5.1 Showing results of triplet extraction for components of the proposed model on NYT and NewsKG21.

Dataset	Model	Precision (%)	Recall (%)	F1 (%)	Correct triplets	Extracted triplets	# Entities
NYT	UDASTE	13.83	13.46	13.64	1160	8387	3675
	No filtering	2.66	14.86	4.51	1280	48190	11128
	Semantic extraction	9.26	4.56	6.11	393	4246	3373
	Restriction extraction	18.60	12.25	12.25	787	4232	829
NewsKG21	UDASTE	20.90	13.05	16.07	92	440	310
	No filtering	6.63	16.74	9.50	118	1780	895
	Semantic extraction	30.48	9.08	13.99	64	210	284
	Restriction extraction	11.57	3.97	5.91	28	242	63

The performance of UDASTE is only slightly better than KGSS on the NewsKG21 dataset which is good keeping in mind that the baseline contains empirical rules that specifically aid it on the NewsKG21 dataset. The presented model shows a much stronger generalisability in news based datasets with the performance on the NYT dataset, where it has a considerably higher score than the other models as seen in Table 5.2.

Table 5.2 Showing results of triplet extraction.

Dataset	Model	Precision (%)	Recall (%)	F1 (%)	Correct triplets	Extracted triplets	# Entities
NewsKG21	OpenIE	4.01	13.62	6.19	96	2395	2184
	KGSS	15.68	16.45	16.06	116	740	702
	UDASTE	20.90	13.05	16.07	92	440	310
WebNLG	OpenIE	0.32	1.63	0.53	26	8175	4268
	KGSS	0.34	0.81	0.48	13	3848	1789
	UDASTE	0.57	1.25	0.78	20	3502	680
NYT	OpenIE	0.67	3.24	1.12	279	41360	30905
	KGSS	3.31	4.21	3.71	363	10961	7245
	UDASTE	13.83	13.46	13.64	1160	8387	3675

Table 5.3 Showing ratio of triplets to entities extracted.

Dataset	OpenIE	KGSS	UDASTE
NewsKG21	1.10	1.05	1.42
WebNLG	1.92	2.15	5.15
NYT	1.34	1.51	2.28

As seen in Table 5.3 UDASTE has a consistently higher triplet to entity ratio, which means that the knowledge graphs produced by UDASTE will be considerably denser and with the number of entities extracted as seen in Table 5.2 they will also be considerably smaller. Denser meaning more relations per entity. This is even without any type of entity disambiguation, which could lead to considerably denser and smaller graphs than the ones currently produced. The main difference that causes this is that UDASTE removes any entities with entity types that are not present in the predefined relation set.

As both KGSS and UDASTE have similar structures it would be expected that the correctly predicted triplets would be similar, but interestingly only 43 of the predicted triplets are in common between the two models. The triplets in common are in large part what is captured by the empirical rules in KGSS and the restriction-based triplet extraction component in UDASTE. A large part of these triplets are location triplets, such as 'Rome, 'Italy' where the two entities appear adjacent to each other. Comparing further with OpenIE as well in Figure 5.1, one can see that there is considerable overlap between UDASTE and KGSS, but that in large part the three models extract different sets of triplets.

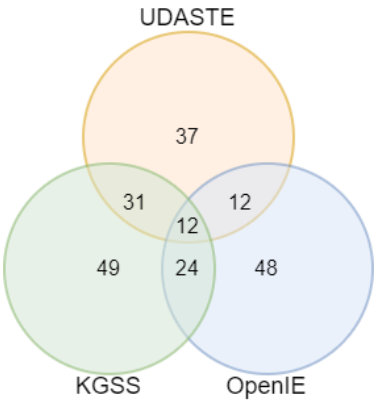


Figure 5.1 Showing the difference in the correctly extracted triplets from NewsKG21 in venn diagram

Regarding what types of triplets the models predict correctly it varies slightly on the dataset, for NYT all three models have location-based relations as the ones they predicted correctly the most often, relations such as 'contains', 'country', and 'neighborhood_of'. Somewhat similarly for NewsKG21 with KGSS and UDASTE the relation 'part_of' is the most correctly predicted, while for OpenIE it is 'place_of_birth'. Again for WebNLG, the predicted triplets are mostly location-based. KGSS seems to have consistently the longest list of relations with at least one correctly guessed triplet. When looking at the distributions of the relations in the annotated triplets, the location-based ones are in general the most common ones, explaining their dominance in the correctly predicted triplets. There are some relations that are underrepresented in the correctly predicted ones, such as the relation 'job_title' from NewsKG21, which is the most common relation in the annotated triplets, but rarely predicted correctly. This is probably due to many of the job titles not being named entities and thus not being extracted. Concept extraction could improve the performance in such cases.

When looking at the wrongly predicted triplets, the large majority for all three datasets is caused by the restriction-based extraction using the apposition tuples. As it predicts one triplet for each allowed entity-type pair and one for each order of the entities in the tuple. Many of the extracted triplets are filtered out, but especially for symmetrical entity tuples, for example, a *(GPE, GPE)* tuple, it becomes a problem as they are not filtered out. Conversely for asymmetrical tuples such as *(PERSON, CARDINAL)* a restriction for the reverse order probably does not exist, meaning it is filtered out. Another relatively common issue is entities getting labelled with the wrong entity type causing wrong relations to be predicted. Thanks to the filters the triplets in general make sense logically, even if their content is wrong, which cannot be said for the two other models.

The performance for all three models is considerably worse for the WebNLG dataset possibly due to its computer-generated nature and fact/statement-type sentences. This does not lend itself to good performance for any of the three models as they all use named entity recognition. For KGSS and UDASTE, an additional reason could be the order of the entities in the triplet is not semantically consistent with the relation. For example, ('Great Britain', 'spokenIn', 'English_Language'), where the opposite order 'English is spoken in Great Britain' would make more sense, or using another relation giving: "In Great Britain they speak English'. The difference is not very large, but testing with the

entities inverted shows the impact it has. It gives very similar performance for KGSS, better for OpenIE, but worse for UDASTE as seen in Table 5.4.

Table 5.4 Showing WebNLG performance evaluated as the other datasets and evaluated by inverting the entities.

Evaluation	Model	Precision (%)	Recall (%)	F1 (%)	Correct triplets	Extracted triplets
Normal	OpenIE	0.32	1.63	0.53	26	8175
	KGSS	0.34	0.81	0.48	13	3848
	UDASTE	0.57	1.25	0.78	20	3502
Inverted	OpenIE	0.51	2.63	0.86	41	8175
	KGSS	0.31	0.75	0.44	12	3848
	UDASTE	0.08	0.19	0.12	3	3502

A possible second reason for worse performance for KGSS and UDASTE on WebNLG is the large number of relations that make it harder for the semantic triplet component to distinguish between them and select the correct one. The filter discussed in 3.5 does also not seem to be very efficient on the WebNLG dataset, with close to 4000 triplets extracted, this again points to a large number of relations being an issue. Especially when many relations share entity pairs, for example with *'is_part_of'*, *'contains'* and *'capital_of'* all containing *(GPE,GPE)* as a possible entity tuple, causing all three relations to be predicted each time two *GPE* entities appear together.

Similarly to what is seen in Table 5.4, the number of triplets where the order of the predicted entities is inverted is relatively large in the NYT dataset as well, this does not seem to be the case in NewsKG21. For KGSS it close to doubles by inverting the entities while for OpenIE it decreases by a factor of 0.4. similarly for UDASTE it decreases by a factor of 0.15, as it better captures the order of the relations. This is in part due to the 'reverse flag', which helps with handling some of NYTs relations. There is a trade-off in the number of permutations of entity pairs extracted that can be predicted versus precision.

Table 5.5 Showing correctly predicted triplets for the NYT dataset, with and without the entity order reversed.

Model	Normal	Inverted
KGSS	363	701
OpenIE	279	107
UDASTE	1160	166

OpenIE has consistently worse performance as seen in Table 5.2. In part due to its tendency to extract an excessive amount of redundant triplets in the scale of 2 times or higher than the other models. The model's recall seems to in general be competitive, with the exception of on the NYT dataset.

During the development of the model, it was decided to compare the embeddings of the predefined relations with the span including the extracted entities, as seen in Figure 5.2 with the arrow marked 'Asymmetrical sentence-relation comparison'. The considered alternatives were to compare the span including the entities with generated sentences using the predefined relations, as seen in Figure 5.2 with the arrow marked 'Full sentence comparison' or to compare only the extracted span excluding the entities with the relations, as marked with 'Relation comparison'.

The Full sentence comparison had a large computational overhead and did not seem to give a large performance improvement, while the asymmetrical comparison seemed to outperform the simpler relation comparison. One possible explanation for the difference in performance is the additional context the entities give to MiniLMs embedding representation.

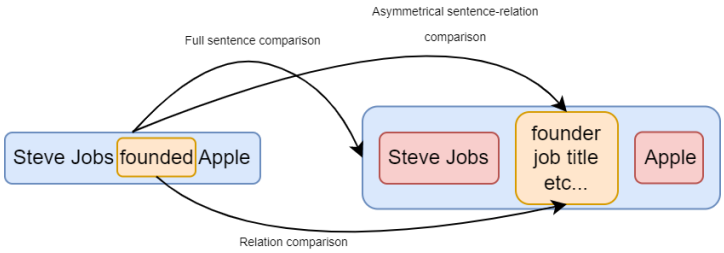


Figure 5.2 Showing the difference between full sentence comparison and Relation comparison

The models components and relative simplicity should allow the model to be relatively flexible regarding uses for other languages as versions of the models used support a range of languages.

6. APPLICATION

UDASTE will be implemented as part of MapIntel which is a competitive intelligence tool that leverages embeddings to search and browse news articles (Silva, 2022). The proposed tools will be implemented as a proof of concept to evaluate the value KGs can provide in a competitive intelligence context. The tool developed will be implemented to aid quick evaluation in terms of the usefulness of news articles. MapIntel's interface consists of an axis where articles are plotted and the distance between articles indicates the similarity of the content between them. What UDASTE adds to this is to create a KG for each individual article, which quickly allows the user to evaluate if the article is of further interest or not. UDASTE is especially apt in that it allows unsupervised triplet extraction where the user can define the set of relations of interest and restrict the type of entities each relation should appear between. This is important as it allows for topical change based on the relation set, in addition to allowing quick iterative changes that can keep up with the daily influx of news articles. The evaluation done with the NewsKG21 and the NYT dataset should be indicative of the performance UDASTE can have in MapIntel tool.

7. CONCLUSION

Information extraction is a challenging task that is dominated by supervised learning methods that are costly to tune, have low flexibility and most of all require labelled data. The existing unsupervised methods have a tendency to over-extract creating a large number of redundant triplets. For triplet extraction models to aid in efficient information extraction a balance between recall and precision is required. The proposed method attempts to limit the number of triplets extracted by using a set of predefined entity types allowed for each relation which allows it to greatly improve its precision. Simultaneously the method avoids using empirical rules in the extraction, by using grammatical structures to extract entity pairs in addition to named entity recognition. The proposed model outperforms two baselines on all three tested datasets, but its performance is considerably better on the news-based datasets. The method although having a relatively low complexity shows promise in applications where the relations can be limited in number and width. Further work on the model could include implementing concept extraction and clause extraction.

7.1. LIMITATIONS

The performance of the model is highly dependent on, the relations used when extracting triplets. The two important facets are the number of relations and the number of allowed entity types for each relation. There is a quite steep decline in F1 score when very broad relations are used, broad relations referring to relations that can be used in a wide variety of contexts. The performance of the system is also highly dependent on the type of sentences used, the WebNLG data contains a lot of facts that don't have named entities or appositions, which leads to considerably lower performance. The model is probably also not very adept at large-scale knowledge graphs as there is no entity disambiguation, so the same entity could appear multiple times with slightly different text which would then appear as two different nodes in a produced KG.

7.2. FURTHER WORK

Using dependency parsing to a larger extent could lead to potentially better results, especially as part of the semantic dependency triplet extraction. It could be used as a method to extract clauses related to two entities. This would allow the model to use all permutations of entities in a sentence, and avoid interference of entities that appear between the two entities at hand. This would be somewhat similar to the clause extraction seen in OpenIE, where a sentence is broken down into a series of clauses that can stand on their own syntactically and semantically (Manning, 2015). Further exploration of grammatical dependencies that lend themselves to triplet extraction could also prove useful. As the model currently stands it can essentially only extract triplets with named entities, an important extension would be to allow the model to extract concepts and the use of restrictions that allow the use of concepts.

8. REFERENCES

An, L. L. (2022). Unsupervised Knowledge Graph Generation Using Semantic Similarity. *Proceedings of the Third Workshop on Deep Learning for Low-Resource Natural Language Processing*, (pp. 169-179).

Boe, B. D. (2014). *Use Cases for.* InterSystems.

diffbot. (2022, 11 29). *diffbot*. Retrieved from diffbot.om

Dyer, G. L. (2016). Neural Architectures for Named Entity Recognition. *Proceedings of the 2016 Conference of the North {A}merican Chapter of the Association for Computational Linguistics: Human Language Technologies.*

Etzioni, A. F. (2011). Identifying Relations for Open Information Extraction. *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing.*

Etzioni, M. M. (2012). Open Language Learning for Information Extraction. *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning.*

Fernandex-Gonzalez, J. N. (2013). Arc-Eager Parsing with the Tree Constraint. *Computational Linguistics*.

Gurevych, N. R. (2019). Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. *arxiv*.

Ives, S. A. (2007). DBpedia: A Nucleus for a Web of Open Data. *International Semantic Web Conference.*

Johnson, M. H. (2015). An Improved Non-monotonic Transition System for Dependency Parsing. *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing.*

kbpedia. (2022, 11 29). *kbpedia*. Retrieved from kbpedia.org

Li, J. L. (2022). A Survey on Deep Learning for Named. *IEEE Transactions on Knowledge and Data Engineering*.

Manning, M. J. (2015). Leveraging Linguistic Structure For Open Domain Information. *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).*

McCallum, S. R. (2010). *Modeling Relations and Their Mentions without.*

Perez-Beltrachini, C. G. (2017). Creating Training Corpora for NLG Micro-Planning. *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers).*

Radford, A. a. (2019). Language models are unsupervised multitask learners. *OpenAI blog*.

Rekatsina, A. G. (2020). Unsupervised Relation Extraction from Language Models using Constrained Cloze Completion. *arxiv*.

Sandhaus, E. (2008). *The New York Times Annotated Corpus.* Abacus Data Network.

Silva, D. F. (2022). Mapintel: enhancing competitive intelligence acquisition through embeddings and visual analytics.

Singh, A. B. (2021). RECON: Relation Extraction using Knowledge Graph Context in a Graph Neural Network. *CoRR*.

Singh, N. S. (2022). DeepSpacy-NER: an efcient deep learning model for named entity. *Evolving Systems*.

Singhall, A. (2022, 11 29). *Introducing the Knowledge Graph: things, not strings*. Retrieved from blog.google: https://blog.google/products/search/introducing-knowledge-graph-things-not/

Song, C. W. (2020). language models are open knowledge graphs. *arxiv*.

Song, C. W. (2021). Zero-Shot Information Extraction as a Unified Text-to-Triple Translation. *arxiv*.

Suchanek, F. a. (2007). Yago: A Core of Semantic Knowledge Unifying WordNet and Wikipedia. *16th international conference on World Wide Web.*

Taylor, K. B. (2008). Freebase: A Collaboratively Created Graph Database For. *SIGMOD '08.*

Toutanova, J. D.-W. (2018). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *Proceedings of the 2019 Conference of the North {A}merican Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers).*

Ulges, M. E. (2020). Span-based Joint Entity and Relation Extraction with. *24th European Conference on Artificial Intelligence - ECAI 2020.*

Vaswani, A. a. (2017). Advances in Neural Information Processing Systems. *Advances in Neural Information Processing Systems.* Curran Associates, Inc.

Wang, W. a. (2020). MINILM: Deep Self-Attention Distillation for Task-Agnostic Compression of Pre-Trained Transformers. *Proceedings of the 34th International Conference on Neural Information Processing Systems.*

Webber, J. B. (2021). *Knowledge Graphs: Data in Context for Responsive Businesses.* O'Reilly Media.

Xiang, X. C. (2020). A review: Knowledge reasoning over knowledge graph. *Expert Systems with Applications*.

Zhan, Q. a. (2018). A Loan Application Fraud Detection Method Based on Knowledge Graph and Neural Network., (pp. 111-115). Shanghai.

Zhao, X. Z. (2018). Extracting Relational Facts by an End-to-End Neural Model with Copy Mechanism. *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers).*

NOVA Information Management School
Instituto Superior de Estatística e Gestão de Informação

Universidade Nova de Lisboa