NLP Project 2 Interim Report

Mark Žakelj[1], Luka Keserič[2], Aleš Kert[3]

Abstract

We are tackling the problem of relationship extraction in definitions. We did not assume the sentences are already segmented into multiple segments. We focused on finding the phrase which is being defined (definiendum) and its hypernym (genus). We used BERT based token classification network to achieve our goal.

Keywords
BERT, Relationship extraction, Hypernym-hyponym

[1] *mz5635@fri.uni-lj.si, 63170037*
[2] *lk6760@fri.uni-lj.si, 6321xxxx*
[3] *ak8754@fri.uni-lj.si, 63170010*

Introduction

Relationship extraction is a problem in natural language processing that aims to find relations between different concepts presented in a corpus. One of these relations, which we will cover in this intermediary report, is the hypernym-hyponym relation, which describes the hierarchical relationship between two concepts. The result of this is a directed graph of concepts, also known as a knowledge graph.

A typical relation extraction task assumes, two different sequences of tokes are already selected and then tries to find the relation between them. This is not the case in our problem, as we must first find sequences of words (segments), which represent a single relation to the defined word (Definiendum, definitor, genus, Location, size, ...). Once these segments are known, we can find the relationship of one group to another, but our focus is mainly on the relation between Definiendum and other groups, as we are dealing with definitions.

We worked on a sentence level and assumed that every sentence includes a definition and therefore includes a definiendum (the word being defined). Our goal was to find the Definiendum and Genus as these form a Hypernym-Hyponym pair, one of the possible relations.

The corpus used in this report is the Termframe corpus, which contains definitions belonging to the Karst domain.

Related work

Our method is based on sequence labeling similar to Named entity recognition and POS tagging tasks. These techniques have been already used for definition extraction in [1], where Bert based Named entity recognition has been fine-tuned for the task. In similar fashion, [2] has used Roberta to generate embeddings and then train CRF on top of it to predict the most probable sequence of tags.

Data preparation

CSV conversion

The Termframe corpus is presented in the WebAnno format, which is the output of an annotation tool used for linguistic annotations, and for our purposes, we had to convert the corpus into a more readable CSV format. We did that by first applying the algorithm made by Vid Podpečan to the entire corpus of english definitions, which produced a CSV file containing the definiendums, geni and whole sentences for each definition. We further modified this by matching the definiendums and the geni to their positions, and splitting the sentences into tokens. In this new CSV file, each line is a token, with it's respective label. The label can have any of the values 'PAD', 'DFD', 'GEN' or 'O', where 'PAD' is the padding token, and 'O' are other words.

We also created a second CSV file, which also contains the part-of-speech(POS) tags for each token, using the HuggingFace framework's pretrained POS classification model. This was done because we wanted to use this feature in some exploratory models, to see if such an addition helps in classification.

Dataset split

This prepared data was then split 80/10/10, which meant that of the 812 definitions, 652 were added to the training data, 80 to the validation data, and 80 to the test data, which was then used to train our model. This is a temporary measure, since the new annotated definitions have not yet been made available, and in the final stage, the model will be trained on the new data.

Methodology

We approached the problem of hypernym-hyponym relation extraction as a sequence labelling problem, where we're looking to label the part of the definition that belongs to the definiendum, which represents hyponym, and the genus, which represents the hypernym.

This classification was done using the BertForTokenClassification from the HuggingFace framework, that has a linear layer on top of the BERT hidden-state output, which was fine-tuned on two types of data - the tokenized sentence and the POS tags, producing two models we will evaluate in later sections.

Our approach thus generated a list of predicted token tags. Because the BERT tokenizer splits words into subwords, we further had to implement an algorithm that combines the tokens into their original form, which meant combining the tokens containing the '##' string with its previous neighbour.

To acquire the detected definiendums and the geni, these sentences were masked using their predicted tags, leaving us with a collection of words belonging to either class. These were then grouped up by combining all words belonging to the same tag, creating a list of definiendums and geni present in the sentence. These terms were then assigned their own node number, and a directed edge between each definiendum and each genus was added, in that direction.

Results

The results of this approach can be seen in Table 1. It turns out that classifying the definiendum in a definition is a much easier task than identifying the genus, with the F1-score being a third higher. Furthermore, in Figure 1, we can see the knowledge graph generated with our limited test data. As we can see, due to the domain containing very specific terms, there's a lot of weakly connected components that only include 2 terms. We think that if more definitions were included, the larger the graph components would be.

There's also an issue we can spot on the graph in the 'visitor capacity' node on the lower left side of the graph. It is a node that is connected to two other nodes, 'number of visitors to' and 'cave'. There's a missing word between them, that would make this one term, making it 'number of visitors to a cave', so we could improve this by checking if a space between two regions contains a character that could connect them.

On a slightly more unrelated note, as expected, the POS tags alone perform much worse than the tokenized sentences, since we're losing information by transforming words into POS tags, but it still manages to find a almost two thirds of definiendums present in the definition. The idea behind this model is to see if including the outputs of a second model improves the performance in any way. This was included because a teammate(that happens to be writing this part) is curious if combining the hidden states of the two models, either by concatenation or averaging, improves the classification accuracy at all.

	Tokenized sentences		
	Precision	Recall	F1-score
GEN	0.55	0.62	0.58
DFD	0.78	0.79	0.78
Average	0.66	0.70	0.68
	POS tags		
	Precision	Recall	F1-score
GEN	0.24	0.28	0.26
DFD	0.65	0.63	0.64
Average	0.45	0.46	0.45

Table 1. Model performances.

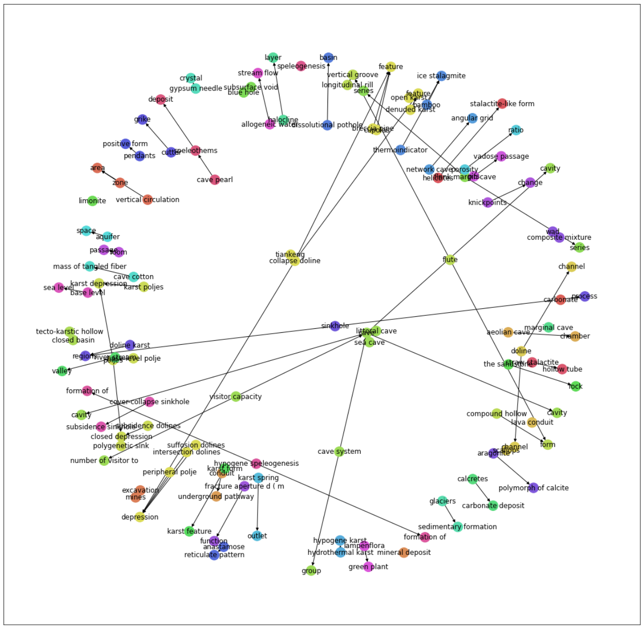

Figure 1. Extracted knowledge graph. This is the graph that we extracted from the test data. It's limited in the number of definitions, but good enough to illustrate a point.

Discussion

As we already discussed, we have a few ideas how we could improve the performance of our model on this task, we will just briefly mention the idea of including POS tags in a second model and combining the output with the model using tokenized sentences, and the idea of finding some way to connect regions that have one space between them, by looking at the words that might appear in that space.

Looking forward at Task 2, where the problem is the extraction of non-hierarchical relations, we think the problem in Task 1 gives us a solid jumping-off point. Looking at the results, we notice that our model is able to identify the definiendum in a definition with a decent accuracy. If we decided on using the R-BERT architecture for determining relations, we could use this model to find the definiendum in a definition, and use this as the first sequence in the model, which would drastically decrease the search space. Additionally, we had an idea to also include a new tag, the region of interest, which would signify tokens in a sentence, that might contain terms the definiendum is related to, further decreasing the search space. If we managed to limit the search space sufficiently, we could then search this region of interest by starting from the beginning and making our way towards the end, and making a cut-off when the probability there exists a relation between two sequences begins to decrease, and continuing until we reach the end of the region.

Another possible technique would be to only find definiendum and then segment the rest of the sentence with *B_region* and *I_region* tags. After sentence segmentation, we could use relation extraction networks like [3] to extract the proper relation.

Acknowledgments

We would like to thank Vid Podpečan for the WebAnno2CSV tool that helped us in setting the data straight.

References

[1] TobiasLee. Defteval2020. https://github.com/TobiasLee/DeftEval2020, 2018.

[2] Andrei-Marius Avram, Dumitru-Clementin Cercel, and Costin Chiru. UPB at SemEval-2020 task 6: Pretrained language models for definition extraction. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, Barcelona (online), December 2020. International Committee for Computational Linguistics.

[3] Livio Baldini Soares, Nicholas FitzGerald, Jeffrey Ling, and Tom Kwiatkowski. Matching the blanks: Distributional similarity for relation learning. *CoRR*, abs/1906.03158, 2019.