Term-tagger development and evaluation

Terminology

Submitted in partial fulfillment
of the requirements for the degree of
MSc Natural Language Processing

by

Soklong Him
Nora Lindvall

Instructed by:
Mathieu Constant

Academic year 2022-2023

Contents

1 Introduction

The aim of this project is to develop and evaluate a term tagger for the domain of Natural Language Processing (NLP). The task is treated like a Name Entity Recognition (NER) task. However, instead of tagging names of people, locations and organisations, we aim to train a model to tag terms specific to the NLP field. The model will be trained using a supervised approach based on an annotated dataset.

1.1 Dataset

The dataset consists of abstracts from scientific papers within the NLP domain from 2009-2021. It was annotated using the IOB format, commonly seen in NER tasks. The annotation was done and corrected by NLP Master students. In total, the dataset contains 965 sentences consisting of 26,665 tokens, out of which 3,256 tokens were tagged as *B* and 3,509 tokens were tagged as *I*. Thus, around 70% of the tokens are tagged as *O*.

Before training a model. The data had to be cleaned as the annotations had many issues. Some annotators used space instead of tab and some used different encodings. Some lines had a label but no token while other lines had a token but no label. Additionally, there were a number of faulty labels, such as "BB" instead of "B" and some single token terms were tagged as I instead of B. This was all corrected with the function *deep_clean* that can be found in the notebook created for this project.

2 Methodology

2.1 Experiment setup

In this project, we will use two diffferent approaches to create an NLP term tagger, namely:

- Conditional Random Fields (CRF)
- Recurrent Neural Network (RNN)

2.2 Conditional Random Fields (CRF)

Conditional random fields (CRF) statistical modelling method that models probability distribution $p(y|x)$, where x is sequence of words and y is sequence of labels (Konkol and Konopík, 2013). CRF extracts features from the text data. The features used in this project include checking whether a token is upper case, lower case or is a digit. In this experiment, we used CRFsuite Okazaki (2007) to implement this algorithm. For training, we used the L-BFGS method for gradient descent.

2.3 Recurrent Neural Network (RNN)

A Recurrent Neural Network (RNN) is an artificial neural network with recurrent connections mostly used in sequence processing. RNN naturally handles variable-length sequences and track long-term dependencies. However, keeping the size of our dataset in mind, the RNN approach may not generate the best results as RNNs generally need at least 10,000 samples for training (Song and Shmatikov, 2019).

Because of our limited data and to improve RNN performance, we decided to use a pretrained model named FastText embedding by Mikolov et al. (2018), which is trained on wikidata.

2.4 Evaluation

As evaluation metrics, we use accuracy, precision, recall and f1 score to compare our two models. Since our goal is to tag NLP terms and the dataset mostly consists of non-terms tagged as O, one could get deceivingly good results if the model tags everything as O. To avoid this, O tags are removed before computing the evaluation metrics.

3 Results & Discussion

After we finished our experiment, we got the result as in Table 1.

Metric	CRF	RNN	RNN (remove O tag)
Accuracy	87.90%	79.30%	44.90
Precision	87.50%	78.40%	85.10%
Recall	87.90%	79.30%	44.90%
F1 score	72.30%	78.80%	57.90%

Table 1: Performance of CRF and RNN model

Based on the resultsa in Table 1, we see that the CRF model performed better than the RNN model. At first glance, the RNN model performed well but it was mostly tagging O-tags correctly. Hence, we evaluate the RNN model without the O tag. It is clear that the O tag has a lot of impact on our model but as the task is to find NLP terms, this is not helpful.

When constructing the RNN model, we tried a few different approaches such as a slice window but it did not improve model performance and was therefore not used.

In conclusion, the CRF model performs better than the RNN model with our corpus. To go further with this experiment one could include POS-tagging, and implementing certain grammatical restrictions. One could also finetune a transformer model like BERT. Ideally, the size of the dataset would be increased and the annotation guidelines clearer.

Bibliography

M. Konkol and M. Konopík. Crf-based czech named entity recognizer and consolidation of czech ner research. In *International conference on text, speech and dialogue*, pages 153–160. Springer, 2013.

T. Mikolov, E. Grave, P. Bojanowski, C. Puhrsch, and A. Joulin. Advances in pre-training distributed word representations. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*, 2018.

N. Okazaki. Crfsuite: a fast implementation of conditional random fields (crfs), 2007. URL http://www.chokkan.org/software/crfsuite/.

C. Song and V. Shmatikov. Auditing data provenance in text-generation models. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 196–206, 2019.