1A-Team / Martin-Luther-Universität Halle-Wittenberg@CLSciSumm 20

Maik Boltze maik.boltze@student.uni-halle.de
Anja Fischer anja.fischer3@student.uni-halle.de
Artur Jurk artur.jurk@student.uni-halle.de
Georg Keller georg.keller@student.uni-halle.de
Lorna Ulbrich lorna.ulbrich@student.uni-halle.de

Abstract

This document demonstrates our groups approach to the CL-SciSumm shared task 2020 (Chandrasekaran et al., 2020). There are three tasks in CL-SciSumm 2020. In Task 1a, we apply a Siamese neural network to identify the spans of text in the reference paper best reflecting a citation. In Task 1b, we use a SVM to classify the facet of a citation.

1 Introduction

Task 1 of the CL-SciSumm shared task 2020 contains two sub tasks. The document dataset for the tasks consists of multiple reference papers (RPs) and for each RP a set of citing papers (CPs) that all contain a citation of the original RP. For each of these citations the cited text spans and the belonging facet have been manually annotated.

For task 1a the goal was to predict the cited text span for a given citation and its reference paper.

In task 1b the participants had to identify what facet a cited text span belongs to, from a predefined set of facets.

Our team's approach utilizes a neural network for task 1a to classify pairs of (citation, reference paper sentence) as either matching or not matching.

For task 1b the syntax of reference text in the form of part-of-speech n-grams is used to predict it's facet.

2 Related Work

Citations play a more significant role in the scientific development than one might expect. Fact is, that they help tracking the development of scientific problems and build a foundation for future research. Citations spread information and are a key attribute of determining the impact of a paper or rather its value to science (Hernández-Alvarez and Gomez, 2016).

There are different methods of extracting useful citations. Some utelize supervised Markov Random Fields classifiers (Qazvinian and Radev, 2010), others modeling the link information and the citation texts (Kataria et al., 2010), or sequence labeling with segment classification (Abu-Jbara and Radev, 2012). The main goal of these approaches is to find the sentences or spans of a CP that explain some facets of the RP. Because a way to see citations is as short textual parts describing some facets of the cited work.

However in this document we don't need to generate or extract citations from a cited work. The citances are already given and we need to find a method to determine the sentence or span in a RP corresponding to the given citance. For this purpose it may help analyzing the aim or rethorical status of a citance like in (Hernández-Alvarez and Gomez, 2016). One work presented a classification framework based on lexically and linguistically inspired features for classifying citation functions (Teufel et al., 2006).

A different mind may think about text summarization as a helping feature to find the corresponding textual span to a given citance. Fortunately the field of summerization grew to a well researched subject in the recent decades. There are several approaches to consider. Some of them are topic modeling (Gong and Liu, 2001), supervised models (Chali and Hasan, 2012), graph based models (Mihalcea, 2004) and neural networks (Chopra et al., 2016). For topic modeling a probabilistic framework is used to estimate the distribution of content in the final summary. Supervised models get a selection of sentences relevant for the final summary to learn on, to afterwards be able to seek the right sentences for a final summary. Graph based models focus on finding the most central sentence in a graph of a text, where sentences are nodes and similarities are edges, which represents a summarizing

Proceedings of the First Workshop on Scholarly Document Processing, pages 277–281

https://doi.org/10.18653/v1/P17

Table 1: Results for Task 1

Task	precision		recall		f1-score	
	micro avg	macro avg	micro avg	macro avg	micro avg	macro avg
1a	0.369	0.403	0.369	0.403	0.369	0.403
1b (POS 5-grams)	0.483	0.482	0.125	0.169	0.199	0.25

Table 2: Structure of input data for task 1a

	citation	original	is_match
0	Another related...	Supersense tagging...	1
1	Another related...	Our approach uses ...	0
2	Another related...	Some specialist to...	0
3	Another related...	Our approach uses ...	0

sentence or to build a summary on.

3 Baseline

Task 1a

As baseline we trained a SVM for each citation and chose the one with the largest tf-idf score as prediction. On the 2018 training set we got an F1-score of 0.09 (micro) and 0.10 (macro).

3.1 Task 1b

The dataset of 2018 consists of a total of 176 citations. 104 citations are labelled as method facet, 9 as implication facet, 34 as result facet, 22 as aim facet and only 7 citations belong to the hypothesis facet. That is why we decided to keep our baseline simple and tagged all citations with the majority label "method". The performance of this simple baseline can be seen in table 4.

4 Approach and Experiment

4.1 Task 1a

Our first preprocessing step is computing the cross product for all citations and every sentence of a reference paper, given annotated citations. The pairs consisting of a citation and its matching reference sentence were labelled as class "1" and all other pairs as class "0". The resulting data matrix, as shown in table 2, contains the citation-sentence pairs and the class labels.

By defining a threshold value of 0.9 we were able to use our NN as a binary classifier. Figure 1 shows the performance of our system when using different thresholds. With our training dataset, a value of 0.9 seemed to be suited best as threshold value.

Our second preprocessing step was mapping each word, which is contained in the word2vec vocabulary (Mikolov et al., 2013b,a) to a unique number in the training data. Based on this, an |word2vec vector size| $\times$ |vocabulary size| embedding matrix E was constructed as a ground layer for the NN. We used a set pre-trained on the GoogleNewsArchive as a word2vec embedding. Reference sentences and citations are represented as one-hot over the vocabulary. Because of the construction of the training data the class "1" was very much underrepresented. For the NN to be able to handle this, we decided to undersample the huge "0" class. This improved our results by a factor of 30, as shown in table 3.

Our system for task 1a is based on a neuronal network (NN) that utilizes two identical long short-term memory (LSTM) networks, mostly referred to as a "Siamese"[1] neural network. The output of the two networks is computed by the exp negate Manhattan distance function (1) as proposed by (Mueller and Thyagarajan, 2016):

$$e^{-||h^{(left)}-h^{(right)}||_1} \quad (1)$$

The complete NN architecture is shown in figure 2.

Table 1 shows the evaluation results of our system on 2017 training data. For the experiment we trained the NN with 2016, 2018 and 2019 training data for 50 epochs and a threshold value of 0.9. We used the "adam" function of the keras tensorflow library (Chollet et al., 2015; Kingma and Ba, 2014) as an optimizer.

4.2 Task 1b

Our approach is based on a support vector machine (SVM) which uses part-of-speech (POS) n-grams as features. During the experiment, we tried using different POS n-gram features in SVMs with linear and polynomial kernels and compared their performances. We did not include the results of SVMs with polynomial kernel, because they showed bad performances.

[1] A Siamese neural network is characterized by using the same weights while working on two different input vectors in tandem, to compute comparable output vectors.

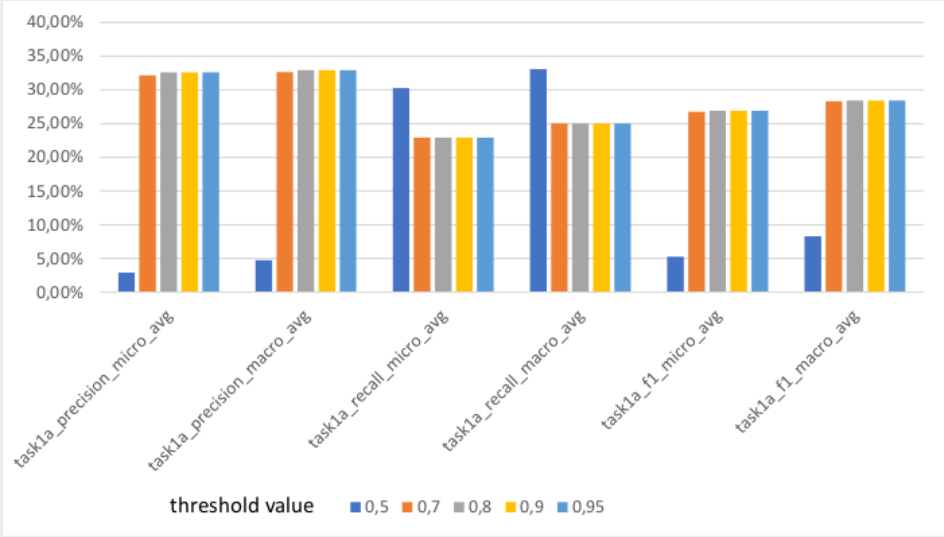

Figure 1: Comparison of threshold values, evaluated on 2017 training data for task 1a

Table 3: Differences between balanced and unbalanced training data for task 1a

	precision		recall		f1-score	
	micro avg	macro avg	micro avg	macro avg	micro avg	macro avg
Without balancing, 25 epochs	0.003	0.003	0.155	0.168	0.005	0.006
With balancing, 25 epochs	0.326	0.329	0.229	0.250	0.269	0.284
With balancing, 50 epochs	0.369	0.403	0.369	0.403	0.369	0.403

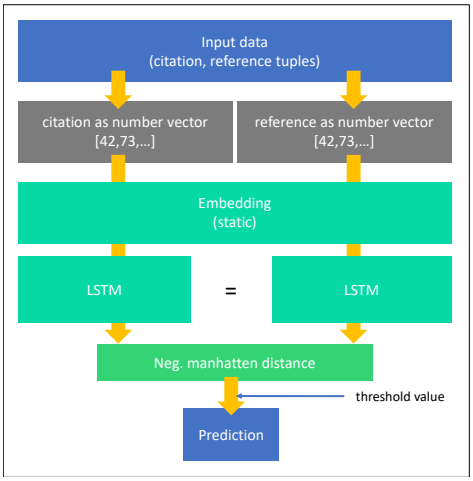

Figure 2: Neural network architecture

In machine learning, kernel methods are a class of algorithms that use a kernel to perform their calculations implicitly in a higher-dimensional space. On one hand we used the function `linear_kernel` which determines the linear kernel. On the other hand we used the function `polynomial_kernel` which determines the degree-d polynomial kernel between two vectors. The polynomial kernel represents the similarity between two vectors.

Basically, the polynomial kernel considers both the similarity between vectors in the same dimension and the similarities across dimensions. When used in machine learning algorithms, this allows to observe the interaction between different features.

The polynomial kernel with input vectors x , y and kernel degree d is defined as:

$$k(x, y) = (yx^T y + c_0)^d$$

If $c_0 = 0$ the kernel is called homogeneous. The linear kernel is a special case of the polynomial kernel where $d = 1$ and $c_0 = 0$. If x, y are column vectors, their linear kernel is described as:

$$k(x, y) = x^T y$$

We tried different degrees with the polynomial kernel, but did not include these in the results of SVMs, because they showed bad performances as well as the results with unbalanced training data. We used the python nltk (Bird et al., 2009) and spaCy (Honnibal and Montani, 2017) libraries for POS tagging and n-gram construction. As shown in table 4 the biggest improvement was gained when increasing n from POS 4-grams to POS 5-grams. Increasing n further seems to deteriorate the results

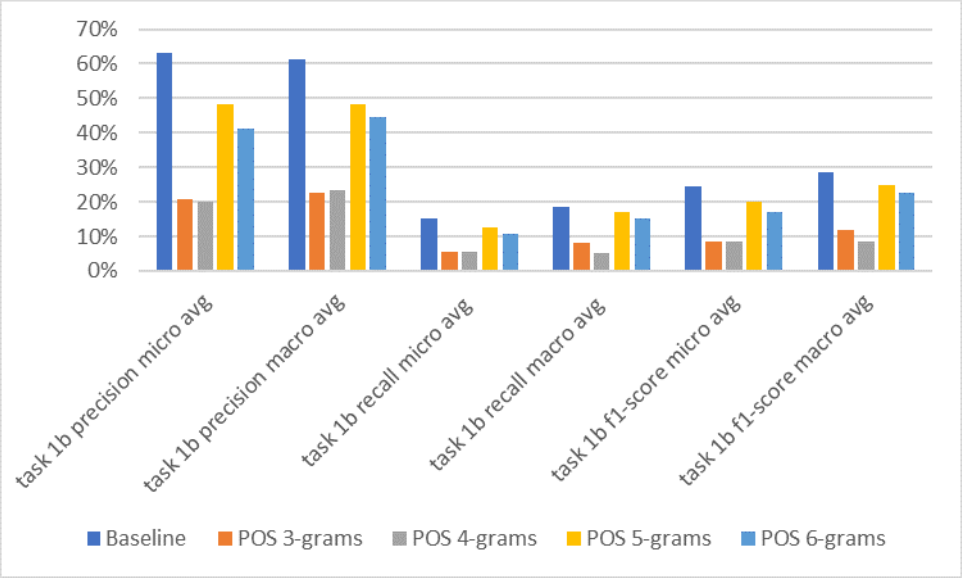

Figure 3: Task 1b results for different POS n-grams

Table 4: Task 1b results for different POS n-grams

	precision		recall		f1-score	
	micro avg	macro avg	micro avg	macro avg	micro avg	macro avg
Baseline	0.63	0.613	0.152	0.186	0.245	0.285
POS 3-grams	0.207	0.226	0.054	0.081	0.085	0.119
POS 4-grams	0.2	0.232	0.054	0.051	0.085	0.084
POS 5-grams	0.483	0.482	0.125	0.169	0.199	0.25
POS 6-grams	0.413	0.446	0.107	0.153	0.17	0.228

again and thus we decided to not test POS n-grams for higher n.

As the results in table 4 and figure 3 show, the best performance was reached using POS 5-grams in combination with a linear kernel SVM.

Conclusion

We could improve upon the solutions of past-year's PolyU approach (Cao et al., 2016) for task 1a. In future works better results may be obtained with more training data as is often the case with neural networks. Moreover the parameters of the neuronal network for task 1a could be tuned.

References

Amjad Abu-Jbara and Dragomir Radev. 2012. Reference scope identification in citing sentences. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 80–90, Montréal, Canada. Association for Computational Linguistics.

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural Language Processing with Python*, 1st edition. O'Reilly Media, Inc.

Ziqiang Cao, Wenjie Li, and Dapeng Wu. 2016. Polyu at cl-scisumm 2016. In *Proceedings of the joint workshop on bibliometric-enhanced information retrieval and natural language processing for digital libraries (BIRNDL)*, pages 132–138.

Yllias Chali and Sadid a. Hasan. 2012. Query-focused multi-document summarization: Automatic data annotations and supervised learning approaches. *Nat. Lang. Eng.*, 18(1):109–145.

M. K. Chandrasekaran, G. Feigenblat, Hovy. E., A. Ravichander, M. Shmueli-Scheuer, and A De Waard. 2020. Overview and insights from scientific document summarization shared tasks 2020: CL-SciSumm, LaySumm and LongSumm. In *Proceedings of the First Workshop on Scholarly Document Processing (SDP 2020)*.

François Chollet et al. 2015. Keras. `https://keras.io`.

Sumit Chopra, Michael Auli, and Alexander M. Rush. 2016. Abstractive sentence summarization with attentive recurrent neural networks. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics:*

Human Language Technologies, pages 93–98, San Diego, California. Association for Computational Linguistics.

Yihong Gong and Xin Liu. 2001. Generic text summarization using relevance measure and latent semantic analysis. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '01, page 19–25, New York, NY, USA. Association for Computing Machinery.

Myriam Hernández-Alvarez and José M. Gomez. 2016. Survey about citation context analysis: Tasks, techniques, and resources. *Natural Language Engineering*, 22(3).

Matthew Honnibal and Ines Montani. 2017. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear.

Saurabh Kataria, Prasenjit Mitra, and Sumit Bhatia. 2010. Utilizing context in generative bayesian models for linked corpus.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization.

Rada Mihalcea. 2004. Graph-based ranking algorithms for sentence extraction, applied to text summarization. In *Proceedings of the ACL 2004 on Interactive Poster and Demonstration Sessions*, ACLdemo '04, page 20–es, USA. Association for Computational Linguistics.

Tomas Mikolov, G.s Corrado, Kai Chen, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. pages 1–12.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'13, page 3111–3119, Red Hook, NY, USA. Curran Associates Inc.

Jonas Mueller and Aditya Thyagarajan. 2016. Siamese recurrent architectures for learning sentence similarity. In *thirtieth AAAI conference on artificial intelligence*.

Vahed Qazvinian and Dragomir Radev. 2010. Identifying non-explicit citing sentences for citation-based summarization. pages 555–564.

Simone Teufel, Advaith Siddharthan, and Dan Tidhar. 2006. Automatic classification of citation function. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 103–110, Sydney, Australia. Association for Computational Linguistics.