Graz University of Technology at CL-SciSumm 2017: Query Generation Strategies

Thomas Felber and Roman Kern

Institute of Interactive Systems and Data Science
Graz University of Technology
Know-Center GmbH
Inffeldgasse 13, 8010 Graz, Austria
felber@student.tugraz.at,rkern@know-center.at

Abstract. In this report we present our contribution to the 3rd Computational Linguistics Scientific Document Summarization Shared Task (CL-SciSumm 2017), which poses the challenge of identifying the spans of text in a reference paper (RP) that most accurately reflect a citation (i.e. citance) from another document to the RP. In our approach, we address this challenge by applying techniques from the field of information retrieval. Therefore we create a separate index for every RP and then transform each citance to a RP into a query. This query is subsequently used to retrieve the most relevant spans of text from the RP. Different ranking models and query generation strategies have been employed to alter which spans of text are retrieved from the index. Furthermore we implemented a k-nn classification based on our search infrastructure for assigning the cited text span to pre-defined classes.

Keywords: Information Retrieval, Query Generation, Ranking Models

1 Introduction

The focus of the CL-SciSumm 2017 Shared Task is on automatic paper summarization in the Computational Linguistics (CL) domain. It is organized as part of the 2nd Joint Workshop on Bibliometric-enhanced Information Retrieval and Natural Language Processing for Digital Libraries (BIRNDL 2017)[1][2], held at the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval[2] in Tokyo, Japan. It is a follow-up on the CL-SciSumm 2016 Shared Task at BIRNDL 2016 [1] which was conducted in the course of the Joint Conference on Digital Libraries (JCDL '16) in Newark, New Jersey.

This Shared Task is divided into multiple smaller tasks which pose the following problems:

– **Task 1A:** Identify the spans of text (cited text spans) in a reference paper (RP) that most accurately reflect a citation (i.e. citance) to the RP made from another document.

[1] http://wing.comp.nus.edu.sg/ birndl-sigir2017/
[2] http://sigir.org/sigir2017/

- **Task 1B:** Classify each identified cited text span according to a predefined set of facets. The elements of that set are: *Implication*, *Method*, *Aim*, *Results*, and *Hypothesis*.
- **Task 2 (optional bonus task):** Generate a structured summary of the RP from the identified cited text spans of the RP, where the length of the summary must not exceed 250 words.

The data set provided for these tasks comprises a training set and a test set consisting of 30 and 10 RPs respectively. Each RP is associated with a set of citing papers (CP) which all contain citations to the RP. In each CP, the text spans (citances) have been identified that pertain to a particular citation to the RP.

To tackle the problem in Task 1A, we followed an information retrieval (IR) approach. For every RP, we created an index holding all the spans of text of that RP. A citance to a RP is transformed into a query and performed on the index associated with the RP to retrieve the most relevant spans of text.

For Task 1B, we followed a k-NN classification approach. Each identified cited text span is compared against all different cited text spans in the training set. Among the top five most similar cited text spans a majority vote is used to determine the facet.

2 Task 1A: Identification of Text Spans in the RP

In this section we provide a closer look at our approach to Task 1A. We will describe how the indices to the RPs are created as well as how the citances are turned into queries and subsequently used to identify relevant spans of text in the RP.

2.1 Index Creation

In order to create an index to a RP, which holds all the different spans of text of the RP, we used the Apache Lucene text search engine library[3] which features Java-based indexing and searching technology.

Taking advantage of the library's indexing technology, we created an index for every RP and added all spans of text of the RP to the index. In this scenario a single span of text can be imagined as a separate text document that is being added to a conventional index. Before we added anything to the index, however, we performed two additional preprocessing steps on every span of text. At first, all stop words contained in a span of text were removed. The idea behind this is to tune the performance of the index (fewer terms in the index) and to obtain more relevant search results since stop words only carry little distinguishing potential [6].

To decide which words qualify as stop words and which do not, we used Apache Lucene's integrated list of stop words for the English language. As a

[3] http://lucene.apache.org/

second preprocessing step, we stripped down all suffixes of all words contained in a span of text in order to normalize them. This was achieved by applying Porter's stemming algorithm [3]. After the preprocessing on a span of text was completed, we moved on and added it to the index.

2.2 Query Generation

After all the indices for the RPs were in place, we transformed each citance to a RP into a query and ran it on the index associated with the RP in order to retrieve the most relevant spans of text of the RP for that particular citance. Since all indices were constructed with Apache Lucene, we also resorted to Apache Lucene functionality to generate the queries. There exists a broad range of different query types in Apache Lucene, however, after we conducted various experiments on the training set, we found that using Apache Lucene's *TermQuery* generated the best results.

To turn a citance into a query, we first applied two preprocessing steps to the citance. The preprocessing steps applied, are analogous to the ones described in section 2, that is, stop words were removed from the citance and porter stemming was performed. As a next step we extracted all words from the citance and created a *TermQuery* for every word. This means, each *TermQuery* corresponds to a single word in the citance. After that, we created an Apache Lucene *BooleanQuery* by OR-conjuncting all *TermQueries*. This resulting *BooleanQuery* was then used to query the index associated with the RP that the citance refers to.

As a result of the query, we obtained a set of top ranked spans of text of the RP. The elements of that set are ordered according to a score, however, it depends on the ranking and retrieval model used by the index what elements are in the set and what score they are given. From all spans of text that are retrieved this way, we considered the top two as most accurately reflecting the corresponding citance. This is because considering the top two yielded the best results during experiments on the training set.

2.3 Ranking

In section 2.2 we mentioned that it depends on the ranking and retrieval model that is used by the index which elements are retrieved by a query and how they are ranked. For the sake of the CL-SciSumm 2017 Shared Task, we submitted a system run using a simple vector space model (VSM) [5] based method and the popular BM25 model [4].

Vector Space Model The term frequency and inverse document frequency (TF-IDF) weighting scheme used by Apache Lucene within the scope of the VSM is as follows:
For the term frequency, which correlates to a term t in a document d, the formula

$$tf_{t,d} = \sqrt{f_{t,d}} \tag{1}$$

is used, where $f_{t,d}$ denotes the number of times the term t occurs in document d.

For the inverse document frequency, which correlates to the number of documents in which the term t appears, the formula

$$idf_t = 1 + \log \frac{N}{n_t + 1} \tag{2}$$

is used, where N is the total number of documents in the index and n_t is the number of documents containing the term t.

The score of a document d to a query q is calculated based on the cosine similarity and is defined as

$$sim(d, q) = \frac{V(d) \cdot V(q)}{|V(d)||V(q)|} \tag{3}$$

where $V(d) \cdot V(q)$ is the dot product of the weighted vectors, and $|V(d)|$ and $|V(q)|$ are their euclidean norms.

BM25 The term frequency factors used in Apache Lucene within the scope of BM25 ranking are defined as

$$\mathcal{B}_{t,d} = \frac{(k_1 + 1) f_{t,d}}{k_1 \left[(1 - b) + b \frac{|d|}{i_{boost}^2 |d|_{avg}} \right] + f_{t,d}} \tag{4}$$

where $f_{t,d}$ denotes the number of times the term t occurs in document d, $|d|$ is the length of the document d in words, $|d|_{avg}$ is the average document length, i_{boost} is an index-time boosting factor, and k_1 and b are parameters.

The ranking equation used in the BM25 model can then be written as

$$sim(d, q) = \sum_{t[q,d]} \mathcal{B}_{t,d} \times \log \frac{N - n_t + 0.5}{n_t + 0.5}. \tag{5}$$

The values we used for the parameters k_1 and b are 1.2 and 0.75 respectively.

3 Task 1B: Identification of the Discourse Facet

The discourse facet takes one of the following values: *Implication*, *Method*, *Aim*, *Results*, and *Hypothesis*. To classify the spans of text, which were identified in Task 1A, we took the following approach: At first we created an index which we filled with all available cited text spans of the training set plus their corresponding discourse facets. To classify which discourse facet a span of text belongs to, we then transformed the span of text into a query, analogous to the way described in 2.2, and then ran the query on the index. After that, we conducted a majority vote on the top five retrieved results to determine the discourse facet to use.

4 Evaluation

Overall we submitted two system runs for Task 1. One of the system runs was conducted using the BM25 ranking model and the other using a simple vector space model (VSM). See section 2.3 for the parameters that we used for these models.

The system performance for Task 1a was determined by measuring the sentence id overlaps between the sentences identified by the system and the gold standard sentences created by human annotators. Based on that, precision, recall and F_1 score were calculated for each system run.

The performance of Task 1b was measured by the proportion of the correctly classified discourse facets by the system, contingent on the expected response of Task 1a. The metrics used here are also precision, recall and F_1 score.

The official evaluation results of our submitted system runs for Task 1a and Task 1b are shown in Table 1 and Table 2 respectively:

Table 1: The Task 1a evaluation results for our system runs using the vector space model (VSM) and BM25.

Ranking Model	**Precision**	**Recall**	F_1 **score**
VSM	0.085	0.138	0.105
BM25	0.107	0.181	0.135

Table 2: The Task 1b evaluation results for our system runs using the vector space model (VSM) and BM25.

Ranking Model	**Precision**	**Recall**	F_1 **score**
VSM	0.917	0.158	0.269
BM25	0.938	0.205	0.337

Judging by the official evaluation results we find that our proposed approaches yield excellent results. Especially our BM25 approach seems to work very well for both Task 1a and Task 1b: An F_1 score of 0.135 at Task 1a achieves the third highest result among all system runs, not far behind the winning system, which has an F_1 score of 0.146. An F_1 score of 0.337 at Task 1b is the eighth highest result among all system runs, with the winning system having an F_1 score of 0.408. Overall 47 system runs have been submitted to Task 1. The mean F_1 score at Task 1a among all system runs is 0.088 and at Task 1b 0.208.

5 Conclusion

In this report we described the approaches we followed to tackle the problems posed in Task 1A and Task 1B of the CL-SciSumm 2017 Shared Task. In preliminary test we found out that using a combination of stop word removal and stemming in combination with a disjunction query strategy work best. The official evaluation results confirmed the success of our approach. We were able to reuse the indexing infrastructure for a classification task, namely assigning categories to the cited text spans. In future work we plan to make use of our infrastructure and investigate methods to enhance the process by integrating more sources of evidence. In particular, additional context information like author or venue specific information might prove beneficial.

Acknowledgements

The Know-Center is funded within the Austrian COMET Program – Competence Centers for Excellent Technologies – under the auspices of the Austrian Federal Ministry of Transport, Innovation and Technology, the Austrian Federal Ministry of Economy, Family and Youth and by the State of Styria. COMET is managed by the Austrian Research Promotion Agency FFG.

References

1. Cabanac, G., Chandrasekaran, M.K., Frommholz, I., Jaidka, K., Kan, M.Y., Mayr, P., Wolfram, D.: Joint workshop on bibliometric-enhanced information retrieval and natural language processing for digital libraries (birndl 2016). In: Proceedings of the 16th ACM/IEEE-CS on Joint Conference on Digital Libraries. pp. 299–300. ACM (2016)
2. Jaidka, K., Chandrasekaran, M.K., Rustagi, S., Kan, M.Y.: Overview of the cl-scisumm 2017 shared task. In: In Proceedings of the Joint Workshop on Bibliometric-enhanced Information Retrieval and Natural Language Processing for Digital Libraries (BIRNDL 2017), Tokyo, Japan, CEUR. (2017)
3. Porter, M.F.: An algorithm for suffix stripping. Program 14(3), 130–137 (1980)
4. Robertson, S.E., Walker, S., Jones, S., Hancock-Beaulieu, M.M., Gatford, M., et al.: Okapi at trec-3. Nist Special Publication Sp 109, 109 (1995)
5. Salton, G., Wong, A., Yang, C.S.: A vector space model for automatic indexing. Commun. ACM 18(11), 613–620 (Nov 1975), http://doi.acm.org/10.1145/361219.361220
6. Silva, C., Ribeiro, B.: The importance of stop word removal on recall values in text categorization. In: Neural Networks, 2003. Proceedings of the International Joint Conference on. vol. 3, pp. 1661–1666. IEEE (2003)