LaSTUS/TALN @ CLSciSumm-17: Cross-document Sentence Matching and Scientific Text Summarization Systems

Ahmed Abura'ed[1], Luis Chiruzzo[2], Horacio Saggion[1], Pablo Accuosto[1], and Àlex Bravo[1]

[1] Universitat Pompeu Fabra
Large-Scale Text Understanding Systems Lab, TALN / DTIC
Barcelona, Spain
ahmed.aburaed@upf.edu, horacio.saggion@upf.edu, pablo.accuosto@upf.edu, alex.bravo@upf.edu
[2] Universidad de la República
Facultad de Ingeniería, Instituto de Computación
Montevideo, Uruguay
luischir@fing.edu.uy

Abstract. In recent years there has been an increasing interest in approaches to scientific summarization that take advantage of the citations a research paper has received in order to extract its main contributions. In this context, the CL-SciSumm 2017 Shared Task has been proposed to address citation-based information extraction and summarization. In this paper we present several systems to address three of the CL-SciSumm tasks. Notably, unsupervised systems to match citing and cited sentences (Task 1A), a supervised approach to identify the type of information being cited (Task 1B), and a supervised citation-based summarizer (Task 2).

1 Introduction

Although scientific summarization has always been an important research topic in the area of natural language processing (NLP) [11, 17, 23, 24] in recent years new summarization approaches have emerged which take advantage of the citations that a scientific article has received in order to extract and summarize its main contributions [18, 19, 1]. It is in this context that a series of challenges have emerged to boost research in the area of citation-based scientific text summarization. Notably, in 2014, the BioSumm 2014 evaluation challenge, and in 2014 and 2016 the CL-SciSumm 2016 challenges [5, 7]. The 2017 edition of the CL-SciSumm challenge [6] proposes to address the following shared tasks: given a cluster of n documents where one is a reference paper and the $n-1$ remaining documents are papers containing citations to it:

– **Task 1A**: For each citance in the citing papers (i.e., text spans containing a citation), identify the cited spans of text in the reference paper that most accurately reflect the citance.

– **Task 1B**: For each cited text span, identify which **discourse facet** it belongs to, among: *Aim*, *Hypothesis*, *Implication*, *Results*, or *Method*.
– **Task 2**: Finally, an optional task consists on generating a structured summary of the reference paper with up to 250 words from the cited text spans.

In this paper we report the systems developed at LaSTUS/TALN to participate in CL-SciSumm 2017. They include four unsupervised systems based on sentence similarity for Task 1A, one supervised approach for Task 1B, and one supervised approach for Task 2. The systems for Tasks 1A and 1B follow the approach reported in [2], where state-of-the-art performance was achieved for Task 1A and almost state-of-the-art performance for Task 1B. The approach for Task 2 follows the method described in [22] which, according to official results [7], was one of the top approaches in CL-SciSumm 2016.

2 Transforming the Source Documents into GATE Language Resources

The organizers of the CL-SciSumm 2017 challenge provided training data structured in clusters of reference and citing papers together with manual annotations indicating, for each citance, the text span(s) in the reference paper that best represent the citance, as well as their corresponding facets. The training corpus contains 30 clusters with an average of 17 papers per cluster. For each cluster there are three manually created summaries of the reference paper: the author abstract, a community-based abstract created using citation sentences, and a human abstract created based on information from reference paper and citation sentences. The test set has 10 clusters with 11 documents each, on average.

In order to automatically process the clusters, we created, from the documents in the training and test sets, GATE [13] files that include the information provided in the manual annotations. The files corresponding to reference papers were enriched with annotations covering the text spans being cited (with the information corresponding to citances) and, conversely, in each citing paper annotations were added for the provided citances (with the information corresponding to the cited text spans). The annotations in the citing and reference papers are linked by means of a unique identifier (formed by the concatenation of citance number, reference paper id, citing paper id, and annotator).

Based on these annotations we could easily build pairs of matching sentences *(Citing Paper Sentence, Reference Paper Sentence)* and associate, to each pair, the facet that the annotator considered the citation referred to (see Task 1B).

2.1 Text Processing

The tokenizer, sentence splitter, part-of-speech tagger, and lemmatizer available in GATE's ANNIE[3] component were used to initially process the documents,

[3] `https://gate.ac.uk/ie/annie.html`

which were then further enriched with annotations obtained by means of the Dr. Inventor (DRI) Text Mining Framework [20]. In particular, we obtained a probability score for each sentence of having a particular rhetorical function in the paper's discourse, among: *Approach*, *Background*, *Challenge*, *Outcome* and *FutureWork*. The SUMMA library [21] was used to obtain normalized tf*idf term vectors for the sentences. For each sentence in the reference paper we computed and annotated its similarity to the sentences in the corresponding citances by means of their cosine distance. For this, we used various vectorial representations of the sentences, including SUMMA's tf*idf vectors and pre-trained word embeddings.

3 Matching Citations to Reference Papers

In this section we present the experiments aimed at detecting which sentence(s) in the reference papers most accurately reflect each given citance.

3.1 Word Embeddings Distance

For the first set of experiments we used the Google News embeddings[4] (three million words in 300 dimensional vectors trained using word2vec [15] over a news text corpus of 100 billion words) and the ACL Anthology Reference Corpus embeddings [10] (100 and 300 dimensional vectors trained over a corpus of ACL papers [3]). Words with similar meanings generate vectors that are close in the embeddings space. From these vectors it is possible to create embeddings for larger units such as phrases, sentences or paragraphs. A simple technique for creating text embeddings that has achieved good results in tasks like extractive summarization [8] and semantic classification [26] is to use the average–or centroid–of the words contained in the texts as their vectorial representations. The embeddings thus created tend to keep the proximity relation if the texts they represent have related–close–words.

We built embeddings for each citance in the citing papers by taking the centroid of the embeddings of all the word contained in it. The same procedure was used to build embeddings for each of the sentences of the reference paper. We calculated the cosine distance for each pair of embeddings: one obtained from each sentence in a reference paper and one corresponding to a citance that refers to it. We experimented with different combinations of embeddings: using only Google News vectors, using only ACL vectors (100 or 300 dimensions) and using the concatenation of Google News and ACL vectors (400 or 600 dimensions). We run several test considering as candidates the top two, five, eight and ten sentences from the reference papers most similar to the given citances. Since the evaluation used for Task 1B uses the F score, we aimed at optimizing this metric. We did this by saving 10 clusters from the training data for validation and using 20 clusters for training. The best performance for the validation set was achieved

[4] `https://code.google.com/archive/p/word2vec/`

using the concatenation of Google and ACL-300 vectors and considering the two top candidate sentences from the reference papers.

3.2 Modified Jaccard

We used a modified version of the Jaccard similarity index that takes into consideration the inverted frequency of the word in a corpus instead of just the word occurrences. For this experiment we calculated idf values of word stems using both the training set and an ACL reference corpus of around 4,000 documents. The modified Jaccard similarity between two text spans s_1 and s_2 is defined in equation 1. Our modification assigns greater weight to matching word stems that are infrequent in the corpus, based on the idea that two text spans that share infrequent words are more likely to be semantically related.

$$MJ(s_1, s_2) = \frac{\sum_{t \in s_1 \cap s_2} 2^{idf(t)}}{|s_1 \cup s_2|} \quad (1)$$

3.3 BabelNet Embeddings Distance

BabelNet [16] is an ontology of concepts (synsets) that integrates many resources, including Wikipedia and WordNet. We used a set of BabelNet embeddings [12] containing 2.5 million vectors trained over a corpus of 300 million words tagged with BabelNet synsets.

Using the Babelfy API,[5] we obtained the list of BabelNet synsets associated to each sentence of the corpus and used them to build sentence embeddings analogously as we did with the word embeddings. The BabelNet embeddings include many vectors for each synset (one for each lexicalization). We therefore calculated the centroid of all the vectors associated to each synset to generate its embedding. For this experiment we proceeded analogously to the word embeddings experiment described above: we calculated embeddings for the citances and for the sentences in the reference papers and then selected as candidates the top N sentences according to their cosine distance to the citances. Based on the tests against the validation corpus, the best results were again achieved considering the two sentences from the reference papers most similar to the citances.

3.4 Voting System

We propose a system that leverages the best results obtained by the word embeddings, Modified Jaccard and BabelNet embeddings systems: the top five candidates obtained for each of the systems are first considered and then a voting process chooses as candidates all the sentences that were selected by at least two systems. If no sentence was chosen by at least two, only the top sentence selected by the Modified Jaccard system[6] is returned. Unlike the other systems described

[5] `http://babelfy.org/guide`

[6] Modified Jaccard was chosen as default as it was the system for which the best F-measures were obtained when run independently.

for this task–where a fixed number of candidate sentences are returned–in this case the number of sentences obtained is variable.

Table 1 shows the performance of the results over the validation data. The experiments are word embeddings (WE), Modified Jaccard (MJ), BabelNet embeddings (BN) and the voting scheme (Voting). The best results over the validation corpus are achieved by the voting system.

Table 1. Performance for Task 1A over the validation corpus.

Method	Avg. Precision	Avg. Recall	Avg. F-Measure
WE	0.077	0.116	0.091
MJ	0.120	0.184	0.144
BN	0.083	0.127	0.099
Voting	0.117	0.199	**0.146**

4 Identifying Citation Facets

In this section we present experiments aimed at identifying the facets the cited text spans belong to. We modeled pairs of reference and citance sentences as feature vectors, which we then used to train classification algorithms that determine whether a cited text span belongs to one of the predefined facets. In the next section we describe the set of features generated. For the classification algorithms we relied on implementations included in the Weka machine learning framework [27].

4.1 Features

Sentence Position Features: The sentence position in a paper can inform about the facet the sentence belongs to. For instance, sentences at the end of the document would probably belong to the *Result* facet. We use three features based on the location of the sentence in the reference document:

- Sentence position: the position of the sentence in the reference paper;
- Section sentence position: the position of the sentence in the section;
- Facet position: five binary features indicating whether the sentence is in a section whose title is associated to one of the target facets (e.g., the word "method" would indicate a section dealing with the facet *Method*).

Text Similarity Features: The more similar a text is to another the more likely it is that they will be part of the same facet. We used two different tf*idf vector representations of the sentences produced by the SUMMA library–one based on word lemmas and one on BabelNet synsets–and computed their cosine similarity. We also calculated the Jaccard and Modified Jaccard coefficients for the lemmas, generating a total of four text similarity features.

Rhetorical Category Probability Features: We mentioned in Section 2.1 that the DRI Framework predicts the probability of a sentence being in one of five possible rhetorical categories. Even if they are different from our targeted discourse facets, we believe that these probabilities could be informative for our classification tasks and therefore also included them as features.
Dr Inventor Sentence Related Features: Other features obtained by means of the DRI Framework that we believed could be of use in predicting a sentence belonging to a particular facet include:

- Citation marker: three features to represent the number of citation markers in the reference sentence, citing sentence and the pair of sentences together;
- Cause and effect: two features to represent if the reference or citing sentence participates in one or more causal relations;
- Co-reference chains: three features to represent the number of nominals and pro-nominals chained in the reference sentence, citing sentence and the pair of sentences together.

Scientific Gazetteer Features: We generated a set of features based on Teufel's action and concept lexicon. The lexicon contains 58 lists. Each one is used to produce a feature which is the ratio of words in the sentence matching the list to the number of words in the sentence. The features are computed for the reference sentence, the citing sentence, and their combination, giving rise to 174 features.
Bag-of-word Features: four string features are produced to represent the *bi-gram lemmas*, *POS-tags bi-gram*, *lemmas* and *POS-tags* for the combination of the reference and the citing sentences.

Based on these features we trained classifiers with 1,386 instances distributed as follows: *Aim* (134), *Implication* (150), *Result* (262), *Hypothesis* (32), *Method* (808). Considering the skewed distribution of the *Method* facet, we decided to train two models: one binary classifier to predict whether the instance is a *Method* or not and a multi-class classifier to identify one of the other facets in case it was previously classified as *not-Method*. We evaluated the performance of several classification algorithms including: SMO algorithm for support vector machines (SMO), naive Bayes (NB), K-nearest neighbors (IBk), random committee (RC), logistic regression (LR) and random forest (RF). We performed 10-fold cross validation experiments with the training data in order to decide which algorithm to use. The best results were obtained with the RF algorithm for the binary *Method* classifier and the SMO for the multi-class classifier representing the *non-Method* facet (Table 2).

Table 2. Algorithms used for the two classifiers trained over the described set of features, evaluated with 10-fold cross validation, with their Precision, Recall and F-measure scores.

Classifier	Algorithm	Avg. Precision	Avg. Recall	Avg. F-Measure
Method Facet [Binary]	RF	0.882	0.875	0.873
Other Facets [Multi-class]	SMO	0.921	0.920	0.920

5 Summarizing Scientific Articles

The proposed summarizer is a modified version of our 2016 summarization system [22] with additional features. The approach is based on a patent summarization method [4] which achieved state-of-the-art performance. It is a trainable sentence scoring, sentence ranking and sentence extraction algorithm which optimally combines the contribution of several numerical features to produce sentence scores. The combination of feature-values is linear and the contribution of each feature is learned using Weka's linear regression algorithm. The summarizer relies on information computed by several tools.

- Each token (i.e., lemma) is weighted by its tf*idf, where idf values are computed from training data;
- For each sentence a vector of terms and normalized weights is created using the previously computed weights (SUMMA vectors);
- Using the ACL word embeddings, a vector is created for each sentence in the document–average of the word embeddings of the words in the sentence (ACL vectors);
- Using the Google news word embeddings, a vector is created for each sentence in the document–average of the word embeddings of the words in the sentence (Google vectors);
- Using the sentence vectors (SUMMA, ACL, Google), three centroids are created for the document–each an average of the sentence vectors in the whole document;
- Using the sentence vectors (SUMMA, ACL, Google), three centroids are created for the abstract of the document–each an average of the sentence vectors in the abstract;
- In the citing papers, token frequency and SUMMA, ACL, Google vectors are also computed.

The features to train the linear regression algorithm are described below. Text similarity features are the result of comparing two vectors of the same type (e.g., SUMMA, ACL, or Google) using the cosine similarity function implemented in SUMMA. Therefore three different feature values are always generated. The reference paper features are as follows:

- Sentence Abstract Similarity Scores: the similarity of a sentence vector to the author abstract vectors (for SUMMA, ACL, Google vectors);
- Sentence Centroid Similarity Scores: the similarity (SUMMA, ACL, Google) of a sentence to the document centroid;
- First Sentence Similarity Scores: the similarity (SUMMA, ACL, Google) of a sentence to the vector of the first sentence (e.g., the title of the reference paper);
- Position Score: the SUMMA implementation of the position method where sentences at the beginning of the document have high scores and sentence at the end of the document have low scores;

– Position in Section Score: a score representing the position of the sentence in the section of the document. Sentences in first section get higher scores, sentences in last section get low scores;
– Sentence Position in Section Score: a position method applied to sentences in each section of the document (sentence at the beginning of the section get higher scores and sentences at the end of the section get lower scores);
– Normalised Cue-phrase Score: the total number of cue-words in the sentence divided by the total number of cue-words in the document. We have relied on [25] formulaic expressions to implement our cue-phrase gazetteer lookup procedure;
– TextRank Normalized Scores: the SUMMA implementation of the TextRank algorithm [14] but with a normalization procedure which yields values for sentences between 0 and 1. Each score is computed using a different sentence vector (SUMMA, ACL, and Google).
– Term Frequency Score: we sum up the tf*idf values of all content words in the sentence and the obtained value is normalized to yield a value between 0 and 1 which is computed using the set of scores from the whole document.
– Citation Marker Score: the ratio of the number of citation markers in the sentence to the total number of citation markers in the paper.
– Rhetorical Class Probability Scores: the probability that the sentence belongs to each DRI rhetorical class.

The citing paper features are as follows:

– Citing Paper Maximum Similarity Scores: each reference paper sentence vector is compared to each citance vector in each citing paper to get the maximum possible cosine similarity (for SUMMA, ACL, and Google vectors);
– Citing Paper Minimum Similarity Scores: each reference paper sentence vector is compared to each citance vector in each citing paper to get the minimum possible cosine similarity (for SUMMA, ACL, and Google vectors);
– Citing Paper Average Similarity Scores: each reference paper sentence vector is compared to each citance vector and the average cosine value obtained (for SUMMA, ACL, and Google vectors);

The approach taken to rank sentences is to produce a cumulative value as the weighted sum of features $f_1, ...f_n$ using the following formula:

$$score(S) = \sum_{i=0}^{n} w_i * f_i \tag{2}$$

with S the sentence to score, f_i the value of feature i and w_i the weight assigned to feature i. The SUMMA system is used to score the sentences once the model has been created.

As we stated before, the weights for each feature are learned from training data and although the ideal score to be learn is in principle unknown, we approximated it with training data. By relying on the gold standard summaries–(a) the author abstract, (b) a human-written abstract, and (c) a community-based

Table 3. ROUGE-2 and ROUGE-SU4 results for all configurations.

Method	ROUGE-2			ROUGE-SU4		
	Abstract	Community	Human	Abstract	Community	Human
ACL_abs	**0.2985**	**0.2000**	**0.1907**	**0.2066**	**0.1164**	**0.1347**
ACL_com	0.2164	0.1889	0.1195	0.1656	0.1129	0.1070
ACL_hum	0.0996	0.1163	0.1055	0.0924	0.0681	0.0895
Google_abs	0.2477	0.1870	0.1365	0.1813	0.1045	0.1003
Google_com	0.1032	0.1600	0.0676	0.0914	0.0832	0.0615
Google_hum	0.1443	0.1143	0.0531	0.1201	0.0701	0.0675
SUMMA_abs	0.2402	0.1436	0.1208	0.1526	0.0860	0.0888
SUMMA_com	0.1687	0.1797	0.0975	0.1189	0.0867	0.0765
SUMMA_hum	0.2181	0.1722	0.1516	0.1611	0.1139	0.1121

abstract–we created different target scores. We compared, using cosine similarity, each sentence vector in the reference paper with each vector in the summary and used the *maximum* similarity values as the target score for the reference paper (e.g., *score(S)*) for learning. This method produced nine different functions to learn: SUMMA, ACL, and Google vectors times *abstract*, *community*, *human* summaries. Note that other target functions are possible but we restricted the number of systems to nine given time constraints. The number of instances used to train the linear regression models was 6,372.

5.1 Evaluating the Summarization Models

Before submission, we carried out a preliminary evaluation of the nine models using 20 document clusters for training and eight document clusters for testing (we could not use two clusters due to errors generated when processing some of the documents in them). The evaluation framework adopted was to compare each of the summaries generated by the model (9 models times 8 clusters = 72 abstracts) against each of the summary types given by the organizers: *abstract*, *community*, and *human*. The comparison was carried out using ROUGE-2 and ROUGE-SU4 [9] (following the configuration suggested by the task organizers). Average results are presented in Table 3 where we highlight the best scores.

6 Submissions to the Challenge and Results

We submitted four runs for tasks 1A, each one applying one of the methods described in Section 3, with the results obtained by the *Method/No-Method* Facet Classifier described in Section 4 for Task 1B.

The organizers of the challenge evaluated Task 1A considering the overlap of the selected sentences and also calculating the resulting ROUGE-2 score. For Task 1B they considered the facet classification based on the sentences previously obtained for Task 1A. The results they obtained with the test set are shown in Table 4, where we include our best result–obtained with the voting system– as well as the maximum, mean and minimum scores for all the systems submitted (macro averages).

Table 4. LaSTUS/TALN Task 1 best results vs. minimum, mean and maximum scores

Score	Task 1A Avg F1	Task 1A ROUGE2 F1	Task 1B Avg F1
LaSTUS/TALN	0.1070	0.0912	0.2930
Min. score	0.0205	0.0339	0.0000
Mean score	0.0882	0.0714	0.2080
Winning score	0.1463	0.1142	0.4081

For Task 2 we submitted nine trainable systems corresponding to nine ways of interpreting the gold standard summaries: three vector representations times three gold standard summaries (system names in first column of Table 3). The organizers of CL-SciSumm used ROUGE-2 and ROUGE-SU4 for the evaluation of the results for Task2. In Table 5 we show our results compared to the mean, minimum and maximum results obtained in the challenge.

Table 5. LaSTUS/TALN Task 2 best results vs. minimum, mean and maximum scores

Score	ROUGE-2			ROUGE-SU4		
	Abstract	Community	Human	Abstract	Community	Human
LaSTUS/TALN	0.2974	0.2169	0.1906	0.1635	0.1655	0.1692
Method	SUMMA_abs	ACL_com	ACL_abs	ACL_abs	ACL_com	ACL_com
Min. score	0.0525	0.1203	0.0748	0.0652	0.0918	0.0963
Mean score	0.2374	0.1926	0.1638	0.1500	0.1413	0.1450
Winning score	0.3506	0.2755	0.2038	0.1914	0.1780	0.1740

7 Conclusions and Outlook

In this paper we have presented unsupervised and supervised methods to address the tasks proposed by the CL-SciSumm 2017 challenge. Our sentence-matching approach takes advantage of both discrete vector representations using terms weighted with tf*idf and continuous word representations. It optimizes several sentence similarity metrics also combining them in a voting system. The facet classifier is a SVM trained on the annotated dataset provided. It uses a set of manually engineered features informed by our previous work. Our citation-based summarization system is a linear regression based algorithm which learns to score sentences based on reference paper and citing paper(s) features, several of them based on continuous word vectors. Our long term goal is to contribute to the areas of extraction and summarization of scientific information. Further work we are considering in this direction include the creation of linguistically enriched scientific datasets and their exploitation to improve access to scientific information. We are also interested in exploring other sentence matching and classification approaches, including some based on deep learning techniques.

Acknowledgments

This work is supported by the Spanish Ministry of Economy and Competitiveness under the Maria de Maeztu Units of Excellence Programme (MDM-2015-0502) and by the TUNER project (TIN2015-65308-C5-5-R, MINECO/FEDER, UE).

References

1. Abu-Jbara, A., Ezra, J., Radev, D.R.: Purpose and polarity of citation: Towards nlp-based bibliometrics. In: HLT-NAACL. pp. 596–606 (2013)
2. AbuRa'ed, A., Chiruzzo, L., Saggion, H.: What sentence are you referring to and why? identifying cited sentences in scientific literature. In: Proceedings of Recent Advances in Natural Language Processing - RANLP 2017. Varna, Bulgaria (2-8 September 2017)
3. Bird, S.: The ACL anthology reference corpus: A reference dataset for bibliographic research in computational linguistics (2008)
4. Codina-Filbà, J., Bouayad-Agha, N., Burga, A., Casamayor, G., Mille, S., Müller, A., Saggion, H., Wanner, L.: Using genre-specific features for patent summaries. Inf. Process. Manage. 53(1), 151–174 (2017)
5. Jaidka, K., Chandrasekaran, M.K., Elizalde, B.F., Jha, R., Jones, C., Kan, M.Y., Khanna, A., Molla-Aliod, D., Radev, D.R., Ronzano, F., Saggion, H.: The computational linguistics summarization pilot task. In: Proceedings of TAC 2014 (2014)
6. Jaidka, K., Chandrasekaran, M.K., Jain, D., Kan, M.Y.: Overview of the CL-SciSumm 2017 shared task. Proceedings of the Joint Workshop on Bibliometric-enhanced Information Retrieval and Natural Language Processing for Digital Libraries (August 2017)
7. Jaidka, K., Chandrasekaran, M.K., Rustagi, S., Kan, M.Y.: Overview of the 2nd Computational Linguistics Scientific Document Summarization Shared Task (CL-SciSumm 2016). In: Proceedings of the Joint Workshop on Bibliometric-enhanced Information Retrieval and Natural Language Processing for Digital Libraries (BIRNDL 2016) (2016)
8. Kågebäck, M., Mogren, O., Tahmasebi, N., Dubhashi, D.: Extractive summarization using continuous vector space models. In: Proceedings of the 2nd Workshop on Continuous Vector Space Models and their Compositionality (CVSC)@ EACL. pp. 31–39. Citeseer (2014)
9. Lin, C.Y.: ROUGE: A package for automatic evaluation of summaries. In: Text summarization branches out: Proceedings of the ACL-04 workshop. vol. 8. Barcelona, Spain (2004)
10. Liu, H.: Sentiment analysis of citations using word2vec. arXiv preprint arXiv:1704.00177 (2017)
11. Luhn, H.P.: The automatic creation of literature abstracts. IBM J. Res. Dev. 2(2), 159–165 (Apr 1958)
12. Mancini, M., Camacho-Collados, J., Iacobacci, I., Navigli, R.: Embedding words and senses together via joint knowledge-enhanced training. arXiv preprint arXiv:1612.02703 (2016)
13. Maynard, D., Tablan, V., Cunningham, H., Ursu, C., Saggion, H., Bontcheva, K., Wilks, Y.: Architectural elements of language engineering robustness. Natural Language Engineering 8(2-3), 257–274 (2002)
14. Mihalcea, R., Tarau, P.: TextRank: Bringing order into texts. In: Proceedings of EMNLP-04and the 2004 Conference on Empirical Methods in Natural Language Processing (July 2004)
15. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. ICLR Workshop (2013)
16. Navigli, R., Ponzetto, S.P.: BabelNet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. Artificial Intelligence 193, 217–250 (2012)

17. Paice, C.D., Jones, P.A.: The identification of important concepts in highly structured technical papers. In: Proceedings of the 16th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 69–78. SIGIR '93, ACM, New York, NY, USA (1993)
18. Qazvinian, V., Radev, D.R.: Scientific paper summarization using citation summary networks. In: Proceedings of the 22Nd International Conference on Computational Linguistics - Volume 1. pp. 689–696. COLING '08, Association for Computational Linguistics, Stroudsburg, PA, USA (2008)
19. Qazvinian, V., Radev, D.R.: Identifying non-explicit citing sentences for citation-based summarization. In: ACL 2010, Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, July 11-16, 2010, Uppsala, Sweden. pp. 555–564 (2010)
20. Ronzano, F., Saggion, H.: Dr. Inventor Framework: Extracting structured information from scientific publications. In: International Conference on Discovery Science. pp. 209–220. Springer (2015)
21. Saggion, H.: SUMMA: A robust and adaptable summarization tool. Traitement Automatique des Langues 49(2) (2008)
22. Saggion, H., AbuRa'ed, A., Ronzano, F.: Trainable citation-enhanced summarization of scientific articles. In: Proceedings of the Joint Workshop on Bibliometric-enhanced Information Retrieval and Natural Language Processing for Digital Libraries (BIRNDL) co-located with the Joint Conference on Digital Libraries 2016 (JCDL 2016), Newark, NJ, USA, June 23, 2016. pp. 175–186 (2016)
23. Saggion, H., Lapalme, G.: Concept identification and presentation in the context of technical text summarization. In: Proceedings of the 2000 NAACL-ANLP Workshop on Automatic Summarization. pp. 1–10. Association for Computational Linguistics, Stroudsburg, PA, USA (2000)
24. Saggion, H., Lapalme, G.: Generating indicative-informative summaries with SumUM. Computational Linguistics 28(4), 497–526 (2002)
25. Teufel, S., Moens, M.: Summarizing scientific articles: Experiments with relevance and rhetorical status. Comput. Linguist. 28(4), 409–445 (Dec 2002)
26. White, L., Togneri, R., Liu, W., Bennamoun, M.: How well sentence embeddings capture meaning. In: Proceedings of the 20th Australasian Document Computing Symposium. p. 9. ACM (2015)
27. Witten, I.H., Frank, E., Hall, M.A., Pal, C.J.: Data Mining: Practical machine learning tools and techniques. Morgan Kaufmann (2016)