Scientometrics (2018) 116:1331–1366
https://doi.org/10.1007/s11192-018-2785-8

Citance-based retrieval and summarization using IR and machine learning

Samaneh Karimi[1,2] · Luis Moraes[2] · Avisha Das[2] · Azadeh Shakery[1,3] · Rakesh Verma[2]

Received: 4 October 2017 / Published online: 4 July 2018

Abstract

We consider the three interesting problems posed by the CL-SciSumm series of shared tasks. Given a reference document D and a set C_D of citances for D: (1) find the span of reference text that corresponds to each citance $c \in C_D$, (2) identify the facet corresponding to each span of reference text from a predefined list of five facets, and (3) construct a summary of at most 250 words for D based on the reference spans. The shared task provided annotated training and test sets for these problems. This paper describes our efforts and the results achieved for each problem, and also a discussion of some interesting parameters of the datasets, which may spur further improvements and innovations.

Keywords Citance-based summarization · Structural correspondence learning · Positional language model · Textual entailment

Research supported in part by NSF Grants DUE 1241772, CNS 1319212, DGE 1433817 and DUE 1356705.

✉ Samaneh Karimi
samanekarimi@ut.ac.ir

Luis Moraes
ltdemoraes@uh.edu

Avisha Das
adas5@uh.edu

Azadeh Shakery
shakery@ut.ac.ir

Rakesh Verma
rverma@uh.edu

[1] School of Electrical and Computer Engineering, College of Engineering, University of Tehran, Tehran, Iran

[2] Computer Science Department, University of Houston, Houston, TX 77204, USA

[3] School of Computer Science, Institute for Research in Fundamental Sciences (IPM), University of Tehran, Tehran, Iran

Introduction

The main goal of the CL-SciSumm 2017 Shared Task (Jaidka et al. 2017) was automated summarization of scientific articles from the computational linguistics domain. We are given a *reference document D* to be summarized along with a set of *citances* C_D—each sentence $c \in C_D$ cites document D. The goal is to create a summary of D that is driven by the citances in C_D. While there has been considerable research in single document summarization techniques (Barrera and Verma 2012; Gambhir and Gupta 2017; Verma and Lee 2017)—the task evaluates the role of citances in generating informative summaries of a paper. This is interesting since a citance can give considerable insight about the purpose and content of the scientific article being summarized, from the viewpoint of the person(s) citing the paper.

For CL-SciSumm 2017, the entire shared task has been split into three subtasks. Given a reference document D and a set C_D of citances for D:

- *Task 1A:* For each citance $c \in C_D$, extract the span of reference text,[1] SR(c), that provides the most information about the citation.[2]
- *Task 1B:* Classification of each SR(c) according to a predefined set of facets: Aim, Method, Hypothesis, Implication, and Results.
- *Task 2:* Generate a summary of at most 250 words for D based on all the SR(c)'s ($c \in C_D$).

We participated in the shared task and our initial results are presented in Karimi et al. (2017). In this paper, we expand upon our methods and experiments and study the problems and datasets in more detail. We present several methods for Tasks 1A and 1B and a simple approach for Task 2. We evaluate the proposed methods on a dataset of scientific articles from the Computational Linguistics domain. This dataset contained 30 documents for training and 10 documents for testing. A detailed description of this dataset is given in "Datasets" section.

For Task 1A, to identify SR(c), we experimented with a number of approaches: structural correspondence learning (SCL), positional language model (PLM), and textual entailment (TE) with two entailment systems.

Each approach returns a score for the sentence in D. The sentences in D are then ranked in non-ascending order by their scores and the top three sentences from D are selected as SR(c). It is challenging to extract just three sentences relevant to a citance from the entire document consisting of hundreds of sentences. Therefore, we also used combinations of the basic approaches and a learning to rank approach to retrieve the best set of sentences to construct SR(c).

We employ SCL modeling technique to learn a joint representation of domains represented by C_D and sentences of D. The second method is a positional language model that leverages proximity information of D to modulate relevance, given a citance $c \in C_D$. We also studied the measure of textual entailment existing between citance c and each sentence s from D—a positive entailment between c and s may imply that $s \in \text{SR}(c)$. We ranked the top sentences extracted by the systems to get the most relevant ones. LambdaRank (Burges 2010) appeared to be one of the best ranking algorithms.

For the facet classification task (Task 1B), we present two methods: a *Rule-based method* augmented by WordNet expansion, a *Machine learning based method* using five

[1] A short piece of text from D.

[2] I.e. what specific information has been cited.

classifiers: SVMs, random forests, decision trees, multi-layer perceptron, and AdaBoost. TFIDF features are used to train the classifiers.

Our approach to Task 2 is simply to sort all the sentences in all the SR(c)'s (for all $c \in C_D$) in the order in which they appear in the document, and then truncate to 250 words.

On Task 1A, the performance of our methods differed considerably on the training and test sets. This provided yet another motivation for us to conduct an analysis of the training and test sets. We believe that this analysis is of independent interest as well.

The rest of the paper is organized as follows. "Preliminaries" section presents the definitions and background for the paper. In "Related work" section we present the relevant related work. "Task 1A: Reference span detection", "Task 1B: Facet detection" and "Task 2: Summary generation" sections present our methods for Tasks 1A, 1B, and 2 respectively. "Datasets" section describes the dataset for CL-SciSumm 2017, our analysis of its characteristics, and our results. "Discussion" section gives our perspectives on the results and "Conclusion" section concludes the paper.

Preliminaries

This section gives a brief description of the terms that have been used throughout the paper.

Cosine similarity A similarity measure between two non-zero vectors A and B is given by the cosine of the angle between them, say θ. Equation 1 gives the formula for calculating cosine similarity.

$$Similarity(A, B) = \cos\,\theta = \frac{A \cdot B}{\|A\| \cdot \|B\|} \tag{1}$$

TFIDF Term frequency-inverse document frequency (Manning et al. 2008) is a popular term weighting method used for selecting the important words across a corpus of textual documents. It ranks words by rewarding them based on their frequency in one document and penalizes the words if they appear across all documents. The method is originally directed towards extraction of documents from a corpus as opposed to extraction of sentences from a document.

For the purpose of our task, we adjust the metric to calculate the scores based on sentence-level granularity in a document as opposed to document-level granularity in a corpus. In other words, we use the frequencies of words in a sentence. Thus the "corpus" in our scenario refers to the entire document. The term frequency (tf_{w_i}) refers to the frequency of the word (w_i) in the sentence. The inverse "document" frequency (idf_{w_i}) is calculated using the number of sentences that contain the word (w_i) in the same document. A sentence S is a collection of word frequencies given by Eq. 2, where N is the number of sentences in the document containing S and df_{w_i} denotes the number of sentences containing w_i in that document.

$$S = \langle s_1, s_2, s_3, \ldots, s_n \rangle \quad \text{where} \quad s_i = tf_{w_i} \cdot idf_{w_i} \tag{2}$$

$$idf_{w_i} = \log(N/df_{w_i}) \tag{3}$$

LDA Latent Dirichlet allocation (LDA) (Blei et al. 2003) is a generative method used for topic detection of a document. While topics in a corpus follow a symmetric Dirichlet distribution, terms in the corpus are assumed to follow a multinomial distribution.

Therefore, the parameters learned from the corpus can be used to determine the topic distribution of terms in a corpus—thus generating the topics of the document. We refer to Eq. 4, where LDA is a measure of topic membership of a sentence S to a topic$_i$. The topic membership vector is used to compare with cosine similarity values.

$$S = \langle s_1, s_2, s_3, \ldots, s_n \rangle \quad \text{where} \quad s_i = P(S \in \text{topic}_i) \tag{4}$$

F_1-score F_1-score is the harmonic mean of precision and recall.

Precision is the proportion of correct results among the results that were returned. And recall is the proportion of correct results among all possible correct results. Our system outputs the top 3 sentences and we compute recall, precision, and F_1-score using these sentences. If a relevant sentence appears in the top 3, then it factors into recall, precision, and F_1-score. Whenever we present the F_1-score on a set of documents, we calculate it through micro-averaging, i.e. averaging among all instances, instead of averaging the F_1-score obtained for each document.

SVM Support vector machines (SVMs) (Cortes and Vapnik 1995) is a discriminative classification method. SVM is a supervised classifier used to linearly classify between data instances even in high dimensional spaces. We use support vector machines for our machine learning based approach in facet detection ("Machine learning approach" section). The SVM model was trained on the training set of reference documents and tested on the given Test set. We use the Scikit-learn python library for the implementation.

RandomForest RandomForest (Breiman 2001) constructs a multitude of decision trees. It uses majority voting across the outputs of the individual trees during classification for the final class decision. The decision trees are usually constructed using a random subset of features from the entire feature list. We use the python Scikit-learn library to build our RandomForest classifier for the supervised facet classification in "Machine learning approach" section. In the *Machine learning based method* experiments for Task 1B, the default values of the parameters are used. The number of trees in the forest is 10, *Gini impurity* is used as the function to measure the quality of a split and nodes are expanded until all leaves are pure or until all leaves contain less than two samples.

Decision trees Decision tree (Quinlan 1986) is used as a non-parametric supervised classifier to create a robust model that predicts the class of a test instance by learning simple decision rules inferred from the given set of attributes. Similar to the previous machine learners, we use decision trees for facet classification in "Machine learning approach" section. The Gini impurity is used to measure the quality of a split.

MLP Multi-layer perceptron (MLP) (Bishop 1995) is a supervised neural learning algorithm. It differs from a simple perceptron in that, between the input and the output layer, there can be one or more non-linear layers, called hidden layers. The system learns a pattern using a feedforward network of neurons and a supervised technique called back-propagation, for calculating weights of the connections. The MLP has been used as a classifier for identification of the facet of the reference text span from the given list ("Machine learning approach" section). We use the default architecture of a single hidden layer with 100 neurons.

AdaBoost Adaptive boosting (AdaBoost) (Freund and Schapire 1997) is a supervised *boosting* machine learner, which can be used to combine several 'weak learners' to improve their performance. The final predicted class is the given by the weighted sum of the outputs of the learners used. The final boosted learner often proves to be a strong classifier. AdaBoost is implemented using Scikit-learn python library in our facet classification system. The decision tree classifier is used as the weak learner in our experiments

for facet detection. The maximum number of estimators at which the boosting is terminated is set to 50.

The next section describes the related work, gives an overview of the shared task and the performance of the participating teams, and ranks the documents based on a measure of their difficulty.

Related work

In Moraes et al. (2017), we have provided an extensive review of the literature on citance-based reference span identification. Citations are considered an important source of information in many text mining areas (Elkiss et al. 2008). For example, citations can be used in summarization to improve a summary (Nanba et al. 2000). It is thought that citations embody the community's perspective on the content of the cited paper (Nakov et al. 2004).

In Qazvinian et al. (2013), the authors illustrate the importance of citations for summarization purposes. They made their summaries based on three sets of information including only the reference article; only the abstract; and, only citations. Finally, they showed that citations produced the best results. In another study, Mohammad et al. (2009) also showed that the information from citations is different from that which can be gleaned from just the abstract or reference article. However, there is one caveat, viz., citations often focus on very specific aspects of a paper (Elkiss et al. 2008).

Facet identification is another task tackled by the participating teams in CL-SciSumm shared tasks. In CL-SciSumm shared task 2016, a feature engineering approach is proposed by one of the participating teams (Lu et al. 2016) to solve the problem. They define a set of features including lexical features such as *tfidf*, the similarity between the topic distributions of citation and candidate reference spans, the concept similarity between citation and candidate reference spans using WordNet and sentence importance. Then they apply three different classifiers including Naïve Bayes, decision tree and support vector machine to identify the facet. Decision tree is also employed by another team in CL-SciSumm 2016 (Cao et al. 2016), which uses the *tfidf* vectors as features. We have also used *tfidf* vectors as features in our classification methods for this task. In Pramanick et al. (2017), authors propose a new method based on the cosine similarity between each candidate sentence vector and each facet's bag of words. A different approach to Task 1B is proposed in Ma et al. (2017), which builds a dictionary for each facet and the reference span is assigned to the facet whose dictionary contains any of the reference span words. Neural networks (Prasad 2017), majority voting (Felber and Kern 2017) and convolutional neural network (Lauscher et al. 2017) are some other approaches proposed by CL-SciSumm 2017 participants for Task 1B. In addition to the classification methods, we have also proposed three variants of a rule-based method, which employs WordNet expansion to identify the facets.

CL-SciSumm 2017

We briefly describe the variety of techniques used by the participating teams in the CL-SciSumm 2017 Shared Task (Jaidka et al. 2017). A total of nine teams participated in Task 1, a subset of five teams further submitted runs for Task 2. Based on the CL-SciSumm 2017 overview (Jaidka et al. 2017), the top three best-performing teams for Task 1A were

NJUST (Ma et al. 2017), TUGRAZ (Felber and Kern 2017) and CIST (Li et al. 2017) with the Sentence-overlap F_1 metric. Based on ROUGE F_1 scores, the top three teams for Task 1A were NJUST, TUGRAZ and UHouston.[3] For Task 1B the top three performing teams were CIST, PKU (Zhang and Li 2017), and NJUST. We summarize below the techniques used by the teams: CIST, NJUST, TUGRAZ, PKU, and UPF (AbuRaed et al. 2017), which did well on Task 2.

The CIST system proposed in Li et al. (2017) calculates similarity values, including Jaccard similarity, context similarity, and *idf* similarity, between reference text and citances. The final results for Task 1A are based on a combination of similarity scores using methods like fusion, majority voting, Jaccard Cascade and Jaccard Focused methods. For Task 1B, they explored better features and tried three methods: rule-based, SVM and fusion. For Task 2 they used Determinantal Point Processes with a linear combination of five types of features that had been previously used. A majority voting across multiple distance-based metrics is used for getting the best pairs of relevant citance and reference text pair in the UPF system (Aburaed et al. 2017). The authors use a multi-class classification system for the facet distribution task.

The majority voting results from an ensemble of classifiers (Linear SVM, SVM using radial basis kernel function, Logistic Regression, Decision Tree) is used for identification of reference and citance spans in the NJUST (Ma et al. 2017). The authors maintained a dictionary of related words for each discourse facet. While evaluating Task 1B, a reference sentence is assigned a facet if it contains any word from the dictionary of the particular facet. The proposed system uses bisecting K-means clustering to generate the summaries for a particular reference document.

For Task 1A, the PKU system (Zhang and Li 2017) uses a combination of sentence-level and character-level *tfidf* scores as well as Word2Vec based similarity values as features to a logistic regression classifier. TUGRAZ (Felber and Kern 2017) proposed a query-based retrieval system where the reference spans are treated as an index and citance acts as the query. For a given citance query, the relevant reference text is chosen depending on the results of a ranking algorithm.

Participating systems' performance

In this section, the performance of all participating systems in CL-SciSumm 2017 is reported with the F_1 score. The plots in this section are based on the workshop proceedings reports of the systems' performance (Jaidka et al. 2017). However, note that the papers were reviewed and revised after the workshop for the proceedings, so the reported results in these papers might not match with the results obtained at the competition stage. Note that the performance we report for Task 1B in this section follows the convention of the shared task. For a correct facet classification to count, the system must have retrieved the correct reference span during Task 1A. Thus, Task 1A acts as an upper bound for Task 1B performance.

We examine the top performing systems on each of the subtasks to analyze the current best-performing techniques for that subtask. In Fig. 1, NJUST is the winner followed by TUGRAZ and CIST. NJUST (Ma et al. 2017) used ensemble learning for identification of reference text based on similarity-based, rule-based and position-based features extracted

[3] Some examples on how this difference in rankings can occur for Task 1A with the two different metrics were given in Jaidka et al. (2017).

from the reference text as well as the citance. CIST (Li et al. 2017) also makes use of similarity scores for Task 1A.

For Task 1B (Fig. 2), CIST (Li et al. 2017) performs the best, and they also had a fusion method for this task. A closer examination of their methods is needed to confirm whether the fusion method indeed had the best score for this task. Our reading of their workshop paper was inconclusive on this point. NJUST (Ma et al. 2017) and TUGRAZ (Felber and Kern 2017) are almost similar in their performance on this task. If we look at the summaries of their approaches for this task in Jaidka et al. (2017), the methods do look similar. Both of them have used an index (called dictionary in NJUST) of reference text along with the facets. Then based on the citance words and which facet(s) in the index contains that word, they identify the citance's facet. A deeper examination of their papers is needed to confirm this.

For Task 2 (Fig. 3), CIST (Li et al. 2017) used a combination of pre-processing techniques that included: document merging, sentence filtering, etc., followed by feature extraction using topic modeling (hLDA) and title similarity. In the final step, the system uses Jaccard similarity for redundancy elimination across chosen reference sentences and Determinantal Point Processes for diverse yet structured summary generation.

Reference documents difficulty for Task 1A

Since all teams participated in Task 1A, we now compare the *10 reference documents in the Test Set* based on the teams' performance (F_1 scores) on each reference document on this task.

For this purpose, all systems' runs for each reference document are sorted based on their sentence overlap F_1 scores, then the top-ranked run of each system is selected and used to represent the system's performance on that reference document. The variance of the

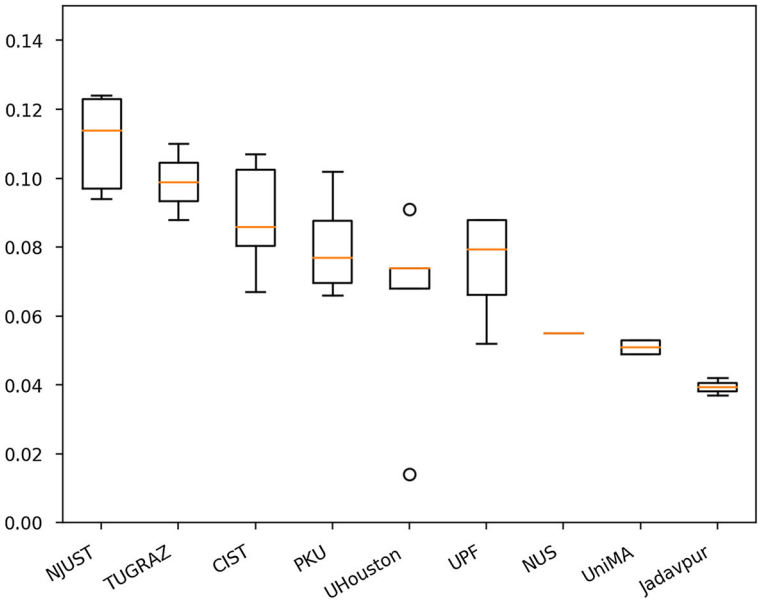

Fig. 1 Systems' performance on Task 1A with sentence overlap F_1 metric

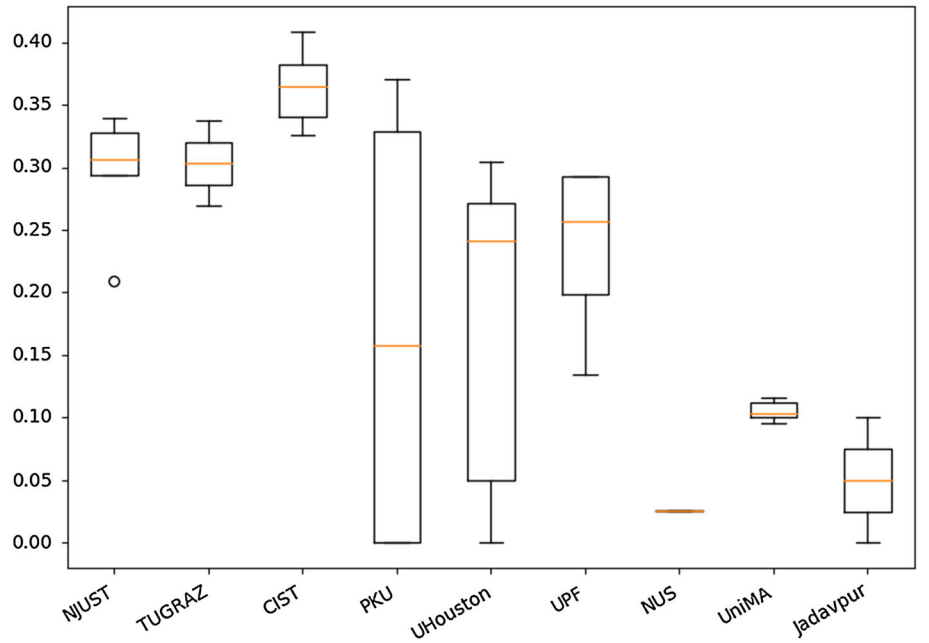

Fig. 2 Systems' performance on Task 1B with F_1 metric

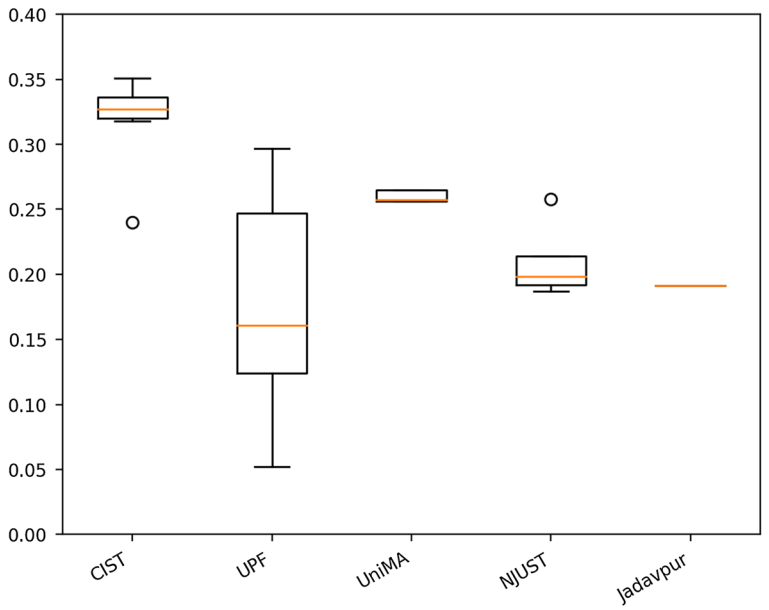

Fig. 3 Systems' performance on Task 2 (summaries vs. abstracts) with ROUGE-2 F_1 metric

Table 1 The variance of the systems' best F_1 scores for each reference document

Reference docs	W11_0815	W09_0621	W06_3909	P07_1040	P00_1025
Variance (σ^2)	0.0046	0.0131	0.0074	0.0010	0.0101
Reference docs	N09_1025	N09_1001	D09_1023	C98_1097	D10_1058
Variance (σ^2)	0.0049	0.0019	0.0131	0.0031	0.0002

systems' best F_1 scores for each reference document is shown in parentheses and also in Table 1. According to Fig. 4, based on the median of each box plot, 'W09-0621' is the easiest reference document in the test set and 'P07-1040' is the most difficult reference document in the test set for the participating systems.

Task 1A: Reference span detection

Our methods for Task 1A include: positional language model, structural correspondence learning, textual entailment and refinements of methods we presented in Moraes et al. (2017) and Moraes et al. (2016). The methods that we refined include TFIDF (Salton and Buckley 1988).

Text preprocessing We employed some pre-processing steps for cleaning the text in the provided datasets for the purpose of our experiments. We remove the contents inside parentheses like names of authors along with removal of special characters[4] (like @, #, etc.), which provide little information in this context. We also skipped reference sentences with no or very little textual content, e.g., sentences comprising one character or word, the result of sentence segmentation in provided datasets.

Positional language model approach

The notion of positional language model (PLM) was used with the goal of retrieving better results in response to a query (with citances being considered as queries and the reference spans as the results of the queries), which employs proximity information in documents in the retrieval process (Lv and Zhai 2009). In this approach, a separate language model is defined for each position (of words) in the document. The PLM of document d at position i is estimated as follows:

$$p(w|d,i) = \frac{c'(w,i)}{\sum_{w' \epsilon V} c'(w',i)},$$

wherein V denotes the vocabulary and $c'(w,i)$ is the propagated count of word w at position i from all of its occurrences in the document.

As shown in the formula above, the weights of the terms in each PLM is estimated based on two factors: their frequencies in the document and also their distance to the position for which the positional language model is built. In other words, the weight of each term in a PLM is the propagated count of that term to that position using a

[4] Only alphanumeric characters remain unfiltered.

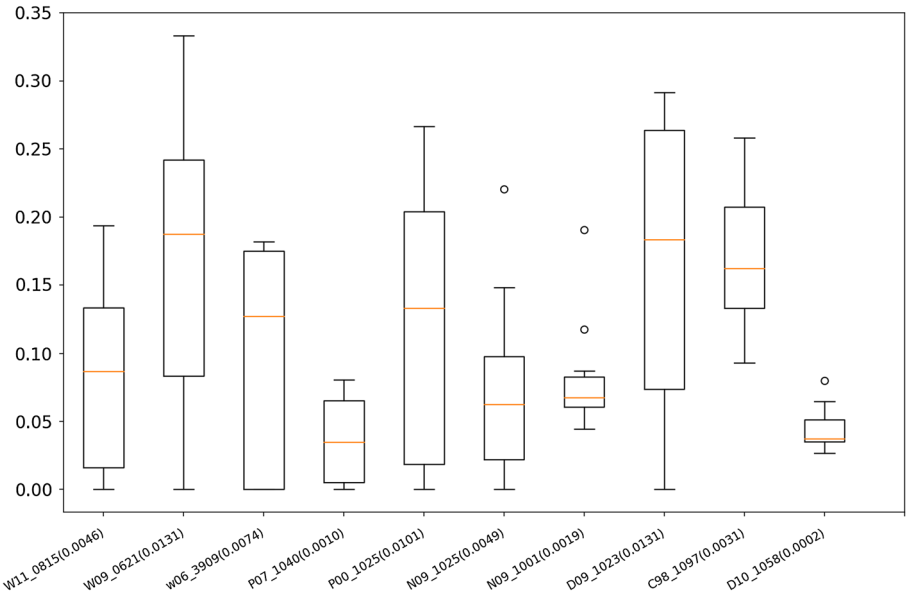

Fig. 4 Participating systems' best runs sentence overlap F_1 score for Task 1A by reference document. The variance is reported in parentheses

propagation function (also called proximity-based kernels) (Lv and Zhai 2009). Gaussian kernel, Triangle kernel, Cosine kernel and Circle kernel are four types of propagation functions that can be used to estimate the terms weights in PLMs. As an example, the following is a Gaussian kernel.

$$k(i,j) = \exp\left[\frac{-(i-j)^2}{2\sigma^2}\right]$$

In other words, in the PLM built for position i, the count of each word, at position j, is weighted by $k(i, j)$. Parameter σ specifies the propagation scope of each word. We used the default value for this parameter.

The total propagated count of word w at position i from the occurrences of w in all the positions is computed according to the following formula:

$$c'(w,i) = \sum_{j=1}^{N} c(w,j)k(i,j)$$

Here N is the length of the document and $c(w, j)$ is the count of term w at position j in the document. In other words, if w occurs at position j, $c(w, j)$ is 1, otherwise 0. After building the PLMs for all of the positions in the document, a position-specific retrieval score can be computed for each position in the document in response to the query by computing the similarity between the language model of the query and the PLM of that position using KL-divergence formula (Kullback and Leibler 1951) which is known as a way of measuring the distance between probability distributions. These position-specific retrieval scores can be used to compute an overall retrieval score for the document through

different strategies. For instance, using best position strategy, the final retrieval score of the document is the score of its best matching position.

We now explain the application of PLM to Task 1A. In Task 1A, each reference sentence is considered as a document and each citance is assumed to be a query. Based on these assumptions, we can use any retrieval method to find the most relevant documents (reference sentences) to the query (citance) to solve Task 1A. For using the PLM approach as a retrieval method, a separate language model is constructed for each position of the reference sentence and then based on the similarity between the positional language models and citance's language model, the similarity score of the reference sentence with the citance can be computed. As mentioned above, the elements of positional language model (PLM) are the propagated counts of all words within the reference sentence which are estimated using a propagation function. With this idea, the closer the words to the position, the higher the weight of the word in the PLM. Therefore, according to the formula above, the PLM of reference sentence *rs* at position *k* is estimated as follows:

$$p(w|rs,k) = \frac{c'(w,k)}{\sum_{w' \epsilon V} c'(w',k)}$$

wherein V denotes the vocabulary of our collection and $c'(w,i)$ is the propagated count of word w at position k from all of its occurrences in the reference sentence. Ultimately, PLM of each position in the reference sentence is compared with the language model of citance using KL-divergence to acquire a position-specific similarity score as follows:

$$S(q,d,i) = -\sum_{w \epsilon V} p(w|q) \log \frac{p(w|q)}{p(w|d,i)}$$

where $p(w|q)$ is the language model of the citance q, $p(w|d, i)$ denotes the positional language model of reference sentence d at position i and $S(q, d, i)$ is the similarity score between the position i in the reference document and the citance. These scores are then used to find the final similarity score of reference sentence (as a document) in response to the citance (as a query). Therefore, we can apply PLM approach as a retrieval process which aims at finding the most relevant reference sentences in response to each citance. The motivation of using PLM for this task is that reference sentences in which the words occurring in the citance appear close to each other are more likely to be relevant to the citance.

In this paper, the PLM implementation released by the authors of Lv and Zhai (2009) is used. In this experiment, the best position strategy is employed for finding the sentence's score based on position-specific scores and a Gaussian kernel is used as a propagation function. Furthermore, positional language models are smoothed using Dirichlet prior smoothing method. Pre-processing steps employed in this experiment are: (1) stopwords are removed from both citances and reference sentences. (2) all special characters are removed, only alphanumeric remain. Parentheses/braces/etc are dealt with differently (by removing internal contents). (3) sentences longer than 70 terms and shorter than eight terms are removed.

Textual entailment approach

Entailment between two pieces of text can be identified as a directional implicational relationship that holds between them. The goal is to measure the degree to which a text fragment can be inferred from another. The pair of text fragments consists of *(a) Text:* The

piece of source information used for drawing the inference and *(b) Hypothesis:* The second fragment, which is to be inferred from the 'text.' The task of deriving inference from pairs of text is called Recognizing Textual Entailment (RTE).[5]

The entailment relationship between a pair of text fragments, i.e., the text (*t*) and the hypothesis (*h*) can belong to one of the following:

- *Positive* When the Text can "prove," i.e., provide strong evidence for, that the Hypothesis is *True*. Thus, the Hypothesis is entailed from the Text. For example, the following pair of text fragments demonstrate positive entailment:

Text (*t*): `The cat ate the fat rat.`
Hypothesis (*h*): `The cat is likely not hungry.`

- *Negative:* When the Hypothesis can be disproved by using the Text. This is an inverse of positive entailment. For example, the following pair demonstrates negative entailment:

Text (*t*): `The cat ate the rat.`
Hypothesis (*h*): `The rat ate the cat.`

- *Neutral* When no relation exists between the two text fragments—the pair is unrelated. For example, the following pair demonstrate neutral entailment:

Text (*t*): `The cat ate the rat.`
Hypothesis (*h*): `The cat and dog are enemies.`

Thus, the property of *textual entailment* between two pieces of text is *True* when the information contained in one text fragment is directly or indirectly derived from the other text fragment.

One of our approaches for Task 1A uses a measure of entailment to extract reference sentences SR(*c*) relevant to a given citance *c*.

In a textual pair used for measuring textual entailment, we use the given citance *c* as (*text*) and a reference sentence *s* from document *D* as (*hypothesis*) to find SR(c). For the calculation of textual entailment, we use two state-of-the-art RTE systems: *TIFMO* (Dong et al. 2014; Tian et al. 2014) and a deep learning model (Zhao et al. 2016).

Textual entailment system A: TIFMO

We use the Textual Inference Forward-chaining Module (TIFMO) (Dong et al. 2014; Tian et al. 2014) to measure textual entailment (TE) between a citance and a sentence from the reference document. TIFMO was chosen as a baseline for our TE approach because it is one of the few systems that is: state-of-the-art, publicly available with good

[5] https://aclweb.org/aclwiki/Recognizing_Textual_Entailment.

documentation, and easy to set up.[6] TIFMO uses Dependency-based Compositional Semantics (DCS) (Tian et al. 2014) trees to represent a text body. The system derives an inference for entailment prediction by considering logic based relations between *'abstract denotations'* or relational expressions generated from the queries in the DCS trees. A further improvement to the system was proposed in Dong et al. (2014), where Generalized Quantifiers (GQs) present in text are taken into account to evaluate lexical and/or syntactical relations between pairs of sentences (text and hypothesis) to predict the presence of entailment and also the type of entailment.

Input generation TIFMO reads the inputs in the form of XML formatted files of text and hypothesis pairs. During evaluation we found TIFMO to be relatively slower (an average of 5 h for processing a set of 100 citance-reference text pairs) compared to our other methods. Therefore, the input to TIFMO (Dong et al. 2014; Tian et al. 2014) was restricted as follows:

(a) We select the top 50 relevant reference sentences (SR'(c), an over-approximation of SR(c)) ranked by our TFIDF system per citance c per document. We used TFIDF since it had the best recall among our systems for Task 1A.
(b) For each given citance c ($c \in C_D$), we generate an XML input file, wherein we have 50 pairs of text fragments (c, s), where c is the text and $s \in \mathrm{SR}'(c)$ is the hypothesis.

TIFMO evaluation TIFMO does not do well as a reference span detector, as seen in Table 5. On the training set of documents, it has recall, precision and F_1 scores of 3.22, 1.68 and 2.21%, respectively. On the test set, it has an F_1 score of 1.41%. We compare the TIFMO results with another deep learning entailment system to check whether the results are specific to TIFMO, or whether the issue is with TE and our problem.

Textual entailment system B: TE using deep learning

A recent trend is for textual entailment systems to make use of deep learning. Systems that use deep learning are usually more robust since they make use of soft alignment schemes. Hence, we also test the performance of a deep learning textual entailment system on our task. The details for the system we employ can be found in Zhao et al. (2016). The authors only provide the code so we had to train our own model. We used the same hyperparameter configuration as the authors to train our model, except we adjusted the learning rate to 0.01 (from 0.001) and the batch size to 128 (from 32). The code was then run on the SNLI corpus (Bowman et al. 2015)—a collection of 570k sentence pairs that were manually labeled for the textual entailment problem. Although the model is better at the textual entailment task, its performance in retrieving sentences relevant to a citance is only marginally better than TIFMO with F_1 scores of 2.87 and 1.69% on train and test, respectively, for sentence overlap.

Our results on using TE systems with Task 1A suggest that recognizing textual entailment has little overlap with our task. This could mean annotators rarely take TE into account when selecting sentences. In order to determine if that was the case, we each manually annotated a sample of the dataset (15 citances from the training set and 5 from the test set). Each annotator is given the list of citances and their corresponding reference spans (SR(c)) in order to determine if entailment occurred. The number of entailments found by each annotator is presented in Table 2. On average, annotators found 5.5 sentences with entailment among the sample of 20, which corresponds to 27.5% of our

[6] The latest version of the software is at https://github.com/tomtung/tifmo.

Table 2 Count of entailments occurring in the sample of 20 citances

	A	B	C	D	Avg.
Entailment count	3	12	6	1	5.5

Table 3 Inter-annotator agreement between A, B, and C

	A/B	A/C	B/C	Avg.	Fleiss' κ
Cohen's κ	0.035	0.305	0.444	0.261	0.194

Table 4 Inter-annotator agreement using a relaxed definition of entailment

	A/B	A/C	B/C	Avg.	Fleiss' κ
Cohen's κ	0.181	0.294	0.528	0.334	0.305

sample. Annotators A, B, and C submitted detailed annotations, thus we report their inter-annotator agreement by way of Cohen's κ (Pontius and Millones 2011) and Fleiss' κ (Fleiss 1971) in Table 3. The moderate to low agreement between the annotators highight the subjective nature of the task. We believe the strict definition of entailment is in large part responsible.

We also asked the annotators to mark the sentences which had *negative* entailment, i.e. entailment in the opposite direction. Once we integrate these negative entailments into the calculations, nearly all inter-annotator agreement scores improve as can be seen in Table 4. Thus, it might be worthwhile to relax the definition of entailment.

Structural correspondence learning approach

Structural correspondence learning is a transfer learning method introduced in Blitzer et al. (2006). Our goal with SCL is to learn how to recognize citations and later transfer this expertise towards recognizing reference spans instead. In order to do so, we must select *pivot features*—these are crucial for the method.

A pivot feature is a feature that is frequent in both domains of interest, such as citances and chosen reference spans. We consider the vocabulary of the union of citances and chosen reference spans that belong to the training set. Words that are frequent in both sets of text are chosen as pivot features. The key to SCL is to predict the occurrence of pivot features from the non-pivot features of an example—we predict the occurrence of a frequent word from the occurrences of the infrequent words in a sentence. For each pivot feature chosen, we learn a different SVM model that predicts whether the pivot feature is present or not, returning a positive or negative label accordingly.

Internally, an SVM has coefficients that determine the importance of a feature for classification. If we collect the coefficients for all the SVM models learned in this manner

we can construct a matrix, which can be used to predict all pivot features simultaneously; a convenient linear algebra trick to run each SVM concurrently.

The next step is to reduce the dimensionality of these predictors—this forces generalization. We apply truncated Singular Value Decomposition (SVD) to the coefficient matrix we constructed previously. The joint representation consists of the predicted pivot features (non-pivot features are thrown away after being used for prediction). For our purposes, these new feature vectors are used to rank candidate sentences through the calculation of cosine similarity scores between them and the citance.

Previous methods

In our previous work, we examined the performance of three different methods: TFIDF, Latent Dirichlet Allocation (Hoffman et al. 2010), and Word Embeddings (Mikolov et al. 2013). Our usage of these methods in this work is changed in one significant respect, which is described below. For a detailed analysis refer to Moraes et al. (2017).

For every citance, we construct a vector where each dimension corresponds to the TFIDF value of a term in the vocabulary of the reference document. Every sentence within the reference document also has its own vector. We compare the vector of every sentence with the citance's vector to determine the sentences with highest similarity. This comparison is performed by calculating the cosine similarity between vectors as explained in the "Preliminaries" section.

One change from Moraes et al. (2017) is that, in addition to unigrams of words, we also considered bigrams and trigrams as part of the terms of a document or sentence. Whenever we refer to a TFIDF system we will also refer to the range of ngrams that the system uses (for instance, 2:3 for bigrams and trigrams excluding unigrams).

For Latent Dirichlet Allocation (LDA), we trained models on a corpus of documents from the ACL Anthology.[7] LDA is a method for topic modeling, so it recognizes a number of topics from the corpus. An LDA model is then used to convert sentences to vectors of topic membership. These are compared with cosine similarity as well.

Finally, we learn word embeddings using the same corpus of documents from the ACL Anthology. However, word embeddings are not as straightforward to use for similarity comparisons. A word embedding will give us a vector for each word in the sentence. One option is to calculate an "average" vector. Instead, we decided to use the Word Mover's distance (Kusner et al. 2015). In essence, given two collections of vectors, we try to align these vectors while moving them as little as possible.

Method combinations

In this section, we explore the potential of method combinations and how best to combine them.

Linear combination

This method was also employed in Moraes et al. (2017). We take the scores from two different systems and generate new scores using the simple formula:

[7] http://aclweb.org/anthology/.

$$\lambda \cdot sys1 + (1 - \lambda) \cdot sys2 \tag{5}$$

To determine the best value for the λ parameter we test different values uniformly along an interval. Whenever we refer to the scores for linear combinations, we shall report only the best-performing system we observed.

Filtering

Another simple way to combine the results from two methods is to rank all the sentences according to one method first. Then, we keep only the top N results and re-rank them according to the second method. We must tune N to achieve the best performance. It is interesting to note that low values of N favor the first system's ranking since the second system is given very little freedom in reordering the top results. All filtering methods had TFIDF as the first system since it was the best individual method on the training dataset. The best filtering system would re-rank the top 5 results from tfidf-1:2 with a word embedding system.

Learning-to-rank

Since we had multiple systems, we opted to combine several through the use of learning-to-rank algorithms. This is a better alternative than trying to tune the previous combination methods for multiple systems. We used a library of learning-to-rank algorithms, RankLib,[8] to combine the scores generated by the other methods. We construct a modified dataset for use with RankLib. For each citance, we construct three different queries by subsampling the irrelevant sentences in the reference document. Therefore, each query consists of all of the relevant sentences chosen by the annotator and 10 irrelevant sentences chosen at random. This helps emphasize learning the ranking of the relevant sentences.

The scores of the following systems were used in conjunction: tfidf-1:1, tfidf-1:2, tfidf-1:3, tfidf-2:3, word2vec (ACL), word2vec (pretrained GoogleNews), variations on LDA, SCL, TIFMO, and deeplearning TE. These systems were chosen in an ad-hoc manner to provide a diverse set of competing rankings. Even though some of these systems underperform in general, they can occasionally provide better rankings for specific citances. No attempt was made to tune the hyperparameters for the algorithms. The learning-to-rank algorithms will attempt to combine the different scorings given by the different systems into a better ranking.

Among the different algorithms implemented by RankLib, many try to minimize an objective function by gradient descent. Sometimes this function is a list-wise cost such as NDCG,[9] in the case of LambdaRank. Other times, the algorithm tries to minimize a simpler function such as pairwise errors, in the case of RankNet. Both MART and LambdaMART are methods based on boosted regression trees.

Since learning-to-rank methods had a considerable jump in performance, we had to test whether overfitting was occurring. We perform tenfold cross-validation on the training set and report the results for a variety of learning-to-rank algorithms. The performance gains measured were much more modest in the cross-validation scenario. In addition, the best learning-to-rank algorithm changed from MART (Burges 2010) to Random Forests.

[8] https://sourceforge.net/p/lemur/wiki/RankLib/.

[9] Normalized Discounted Cumulative Gain is a metric for search results that takes into account the position of relevant items.

Table 5 Task 1A scores for individual systems on the 2017 dataset

Method	Train			Test (%)
	P@3 (%)	R@3 (%)	F_1 (%)	F_1
tfidf-1:1	11.05	21.20	14.53	6.77
tfidf-1:2	11.39	21.85	14.97	7.62
tfidf-1:3	11.05	21.20	14.53	6.77
tfidf-2:3	8.64	16.57	11.36	7.06
Word2vec	10.88	20.88	14.31	9.03
LDA	2.63	5.05	3.46	4.51
PLM	7.29	13.99	9.59	6.21
KL-div	6.84	13.13	9.00	6.21
Okapi	7.85	15.06	10.32	6.21
SCL	3.14	6.02	4.13	1.69
TIFMO	1.68	3.22	2.21	1.41
Deeplearn TE	2.18	4.19	2.87	1.69
Linear Comb.	11.72	22.49	15.41	7.06
Filtering	11.78	22.60	15.49	7.34
Randomforest	17.34 (11.67)	33.26 (22.38)	22.79 (15.34)	6.28
Coordascent	11.67 (11.50)	22.38 (22.06)	15.34 (15.12)	8.47
Rankboost	11.61 (11.27)	22.28 (21.63)	15.27 (14.82)	8.47
Linreg	11.27 (10.99)	21.63 (21.09)	14.82 (14.45)	5.64
Lambdamart	25.75 (9.42)	49.40 (18.08)	33.86 (12.39)	8.47
Mart	26.20 (8.86)	50.26 (17.00)	34.45 (11.65)	8.47
Listnet	2.24 (2.13)	4.30 (4.09)	2.95 (2.80)	1.41
Ranknet	2.24 (1.79)	4.30 (3.44)	2.95 (2.36)	5.64
Lambdarank	5.33 (1.57)	10.22 (3.01)	7.00 (2.06)	0.00

Cross-validated results appear in parentheses

Results

We now report the results on the training and test sets for the methods employed in Task 1A which can be found in Table 5. Precision, Recall, and F1 are three measures used to evaluate and compare the methods. In general, TFIDF still performed well among our systems on the training set. However, once we move to the test set its performance degrades severely. In fact, across the board the performance is worse on the test set.

Looking closer at the discrepancy between the training and test set we can see some interesting behavior: the system based on word embeddings was our most robust system. Although previous work in Moraes et al. (2017) downplayed the importance of word embeddings (because combining them with TFIDF did not improve our performance by much) it seems they may have other advantages. In particular, we suspect the use of the Word Mover's Distance is what led to this robustness.

Two information retrieval methods, KL-divergence and Okapi, are employed to compare with PLM. In all three methods, reference sentences are considered as documents and citances as queries. Okapi is a ranking function that is based on the probabilistic retrieval framework, and KL-divergence is a language modeling retrieval approach, which ranks

documents based on the similarity between their language models and query's language model using KL-divergence formula. According to Table 5 PLM and Okapi perform better than KL-div on training set. However, their performance on test set are the same. The drop in performance on the test set resulted in several ties, which were scrutinized carefully to eliminate the possibilities of errors in evaluation or implementation creating the ties.

Task 1B: Facet detection

In Task 1B, for each SR(c), the facet to which it belongs is picked among a predefined set of five facets. This task involves mainly two approaches: a rule-based approach and a machine learning approach. In both of our approaches, we employ the citance instead of the reference text to identify the facet, based on the assumption that the relation between citances and their reference texts can help the facet identification method find the correct facet.

Rule-based approach

The Rule-based approach is comprised of three sequential steps where each one is designed to find the right facet through specific comparisons, in case no match was found in any of the previous steps. In the first step, citance words are compared with all five facet labels: Method, Implication, Result, Hypothesis and Aim. If none of the words in the citance match a facet label, then we proceed to the second step. In the second step of the rule-based approach, an expanded form of the citance is compared with the facet labels. The citance is expanded by adding all WordNet synsets (Miller 1995) of each word found in the citance. In the last step, if no matched facet label is found within the previous steps, the facet labels are expanded with their synsets and once again are compared with the words in the citance.

Machine learning approach

In this approach, each citance is represented by a feature vector containing TFIDF values of its terms which is the number of times a term occurs in the citance multiplied with idf component which is computed according to Eq. 3. The total number of features used in these experiments is 4663.

After classification model is learned using the training set, the trained model is used to classify citances of the test set. Machine learning methods used in this approach include support vector machines (SVMs) (Cortes and Vapnik 1995), random forests (Breiman 2001), decision trees (Quinlan 1986), MLP (Bishop 1995), and Adaboost (Freund and Schapire 1997).

Evaluation

The rule-based approach has different variations: (1) rule_based-V1: In this variation, all three sets of comparisons (comparing citance words with facet labels, comparing expanded form of citances with facet labels and comparing expanded form of facets with citance words) are done while non-relevant synsets of all facets are excluded. To find the non-relevant synsets, we manually investigate all synsets of each facet label in WordNet and exclude those that seem irrelevant. (2) Rule_based-V2: in the second variation, all three sets of comparisons are done while only non-relevant synsets of "Method" facet are

Table 6 Recall, precision, and F_1 score of rule-based method variations (Task 1B)

Method	Train			Test
	P (%)	R (%)	F_1 (%)	F_1 (%)
Rule_based-V1	47.82	42.30	44.89	28.84
Rule_based-V2	63.24	55.94	59.36	68.33
Rule_based-V3	68.37	60.48	64.19	78.99
Method_only	69.16	61.18	64.93	95.29

Table 7 Recall, precision, and F_1 score of classification methods (Task 1B)

Method	Train			Test
	P (%)	R (%)	F_1 (%)	F_1 (%)
SVM	99.38 (63.58)	98.92 (58.67)	99.15 (60.98)	73.35
Random Forest	92.85 (62.94)	92.14 (57.62)	92.49 (60.11)	72.50
Decision Tree	98.46 (48.67)	98.76 (53.55)	98.61 (50.88)	56.89
MLP	100.0 (57.38)	100.00 (54.30)	100.00 (55.72)	65.83
Adaboost	83.99 (46.09)	88.13 (47.21)	86.01 (46.60)	61.72
Rule_based-V1	47.82	42.30	44.89	28.84
Rule_based-V2	63.24	55.94	59.36	68.33
Rule_based-V3	68.37	60.48	64.19	78.99
Method_only	69.16	61.18	64.93	95.29

Cross-validated results appear in parentheses

excluded. (3) Rule_based-V3: in the third variation, only first and second comparisons are done. Table 6 represents the results of the first approach to Task 1B on training set 2017 and test set 2017. A "Method_only" approach which assigns "method" to all of the citances is also employed to be compared with the rule-based approach.

As Table 6 shows, the third variation of the rule-based approach outperforms other variations on both training and test sets. It means that expansion of facet labels does not help in finding the correct facet label of reference spans when citances are used for the computation.

Furthermore, the higher performance of Rule_based-V2 over Rule_based-V1 shows that excluding non-relevant synsets of the "Method" facet leads to better results. It might be due to the fact that "Method" is the most frequent facet label in both the training and test set for 2017. The results of the Method-only approach also verify this fact.

Table 7 shows the results of Task 1B for machine learning methods on the training and test set. For classification experiments on the training set, two set of results are reported in Table 7: (1) the results which are obtained by training the classifier using the whole 30 documents of the training set and testing on the same set of 30 documents as test data (similar to the rule-based methods results) and (2) the tenfold cross-validation results, in parentheses. For the classification experiments on the test set, the whole training set is used for the learning phase.

As Table 7 shows SVM has the best performance in comparison with other classification methods on Task 1B and the lowest results among classification methods belong to the Decision Tree. Furthermore, comparison between the results of Tables 6 and 7 shows that Rule_based-V3 is our best-performing method on Task 1B among all rule-based and classification methods.

Table 8 ROUGE-2 scores for summarization methods (Task 2) on the 2017 test set

Method	Avg. precision (%)	Avg. recall (%)	Avg. F1 (%)
Word2vec	21.84	27.73	24.40
tfidf-1:1	21.18	24.93	22.88
tfidf-1:3	20.15	24.30	21.98
tfidf-1:2	19.81	24.11	21.70
Lambdamart	19.45	23.85	21.38
Filter	18.32	23.06	20.39
Lambda	26.77	16.39	20.27
tfidf-2:3	17.38	22.73	19.66

Task 2: Summary generation

In summary generation experiments, the reference span detection results of five best-performing methods on the training set and five best-performing methods on the test set are chosen to be used for evaluation of the summarization task. Table 8 includes the union of these two sets of methods which includes five method due to the duplicates in two sets of best-performing methods. The summary of each reference document via each method is extracted from reference spans detected by the method, which is cut off according to the summary's length limit which is 250 words.

In Table 8, average precision, recall, and F1 scores are reported for the five best-performing methods on the 2017 test set using the ROUGE toolkit (Lin 2004), specifically ROUGE-2 which counts the overlap of bigrams between the system generated summary and the gold standard. According to Table 8, word embeddings outperform other methods in summarization.

Datasets

In this section, we investigate the differences between the training set and the test set that could possibly account for the loss of performance when going from one to the other. We believe the lower performance can be explained by a larger percentage of challenging instances. First, we review the quantitative characteristics of each set. In addition to statistics such as word counts and facet distributions, we also compare various metrics that try to capture qualitative assessments, such as reading difficulty. Overall, our goal is to determine whether there are metrics that can recognize challenging instances.

Dataset statistics

The dataset for CL-SciSumm 2017 (Jaidka et al. 2017) contains 30 training documents and 10 testing documents, each with multiple citances. We use Scikit-learn to tokenize the sentences. Some statistics, without any preprocessing, about the *training dataset* (30 documents) are reported below.

- The total number of sentences is 6700 across all documents and the average is 223.33 per document.
- The total number of citances is 594 across all documents and the average is 19.8 per document.

- There are 529 unique sentences chosen as SR(c) across all documents and the average is 17.63 per document.
- The set contains 139,842 words, of which 121,291 are unique, among the reference documents.
- The average number of words per document is 4661.4 and the standard deviation is 2546.7 words.

Some statistics about the *test dataset* (10 documents), also without any preprocessing, are as follows:

- The total number of sentences is 2012 across all documents and the average is 201.2 per document.
- The total number of citances is 159 across all documents and the average is 15.9 per document.
- There are 152 unique sentences chosen as SR(c) across all documents and the average is 15.2 per document.
- The set contains 40,558 words, of which 10,313 are unique, among the reference documents.
- The average number of words per document is 4055.8 and the standard deviation is 1101.3 words.

The results of the reference span detection and facet detection methods are quite different on the test set from those on the training set. We analyze this difference by comparing the datasets' characteristics including their lingual or statistical characteristics.

Facet distribution

We evaluate the gold standard annotation files of all citances, c, and reference spans, SR(c), to observe their distribution across the set of predefined facets: Aim, Method, Hypothesis, Implication, and Results. Tables 9 and 10 give details of the distribution across facets for training and test documents.

We observe that there is a significant variation in the distribution of citances (c) across the facets in training data compared to test data.[10] While approx. 96% of the citances $C_{D,\text{test}}$ extracted from the test documents belong to the Method facet, around 69% of the training set citances $C_{D,\text{train}}$ are in the Method facet. Also, while the training set has citances which belong to Implication and/or Hypothesis facets, test set has 0 citances belonging to Hypothesis or Implication facets. Thus the facet distribution across all $C_{D,\text{train}}$ is clearly unbalanced with respect to corresponding test set citances $C_{D,\text{test}}$.

Text difficulty level

In this section, we study and characterize the "difficulty" of the test documents versus that of the training set. For this purpose, Flesch–Kincaid grade level (Kincaid et al. 1975), SMOG readability index (McLaughlin 1969) and Gunning's FOG index (Dubay 2004) are employed as three text difficulty measures to compare test set and training set documents. All of these three measures are calculated using textstat 0.4.1, which is a Python package for calculating statistical features from text (Shivam Bansal 2017).

[10] Recall that we use the citances to solve the Task 1B.

Table 9 Distribution of reference span facets by citances in training documents of CL-SciSumm'17

TrainSetIDs	Method	Aim	Result(s)	Hypo.	Implic.	Total Cit.	Total Facets
C00-2123	17	1	0	0	2	18	20
C02-1025	12	0	5	0	1	18	18
C04-1089	13	6	0	0	2	15	21
C08-1098	21	1	6	0	2	28	30
C10-1045	14	6	12	0	0	31	32
C90-2039	6	2	1	0	4	13	13
C94-2154	3	0	0	1	0	4	4
D10-1083	10	2	5	0	0	16	17
E03-1020	11	0	1	0	1	13	13
E09-2008	8	0	0	0	0	8	8
H05-1115	3	8	0	0	0	11	11
H89-2014	8	1	0	0	1	10	10
I05-5011	13	1	0	0	5	17	19
J00-3003	6	4	0	0	0	10	10
J96-3004	42	1	13	0	10	64	66
J98-2005	1	3	0	0	0	4	4
N01-1011	3	0	2	0	1	6	6
N04-1038	17	5	1	0	2	22	25
N06-2049	16	0	5	0	2	20	23
P05-1004	13	0	0	0	0	13	13
P05-1053	37	1	12	1	8	56	59
P06-2124	9	3	3	5	3	15	23
P98-1046	8	1	2	8	3	21	22
P98-1081	12	1	7	0	1	21	21
P98-2143	43	1	7	0	5	49	56
W03-0410	15	1	3	1	2	21	22
W04-0213	15	0	0	0	1	15	16
W08-2222	7	0	0	0	2	8	9
W95-0104	25	12	9	2	0	37	48
X96-1048	4	0	4	0	2	10	10
Total	412	61	98	18	60	594	649
Total (%)	69.36	10.27	16.50	3.03	10.10	100.00	

In this experiment, Flesch–Kincaid grade level, SMOG readability index and Gunning's FOG index are computed for citances and their corresponding reference text of each reference document separately. Tables 11 and 12 show the results of this experiment on Training set and Test set respectively.

In the next experiment, the three measures are computed for unsolved-by-us citances.[11] As citances in this problem act as queries in Information Retrieval problems, the higher difficulty level of citances of the test set in comparison with the training set can be considered as the reason for the lower performance of our methods on the test set rather

[11] Citances for which all our systems failed to identify the correct reference spans.

Table 10 Distribution of reference spans facets by citances in test documents of CL-SciSumm'17

TestSetIDs	Method	Aim	Result(s)	Hypo.	Implic.	Total Cit.	Total facets
C98-1097	12	0	0	0	0	12	12
D09-1023	10	0	2	0	0	12	12
D10-1058	18	0	0	0	0	18	18
N09-1001	11	1	2	0	0	13	14
N09-1025	32	0	0	0	0	32	32
P00-1025	11	0	1	0	0	12	12
P07-1040	26	0	0	0	0	26	26
W06-3909	12	0	0	0	0	12	12
W09-0621	12	0	0	0	0	12	12
W11-0815	8	0	2	0	0	10	10
Total	152	1	7	0	0	159	160
Total (%)	95.60	0.63	4.40	0.00	0.00	100.00	

than the training set. After computing the difficulty measures on unsolved-by-us citances of the test and training sets, a Mann–Whitney (Mann and Whitney 1947) test is used as a non-parametric statistic to test if the difference between difficulty measure values on unsolved-by-us citances of test set and training set are statistically significant. For our implementation, we use the SciPy Python library—scipy.stats (Jones et al. 2001–), for statistical evaluations. Using the Mann–Whitney test, we can check if there is a statistically significant difference between two data groups without any constraints on the data to be normally distributed. This test works by ranking the data in each group and computing the mean ranks for each of them. If the distributions of the groups are identical, the mean rank will be the same for both groups (Mann and Whitney 1947). The null hypothesis for the Mann–Whitney test in this experiment is that the distribution of difficulty measure values on unsolved-by-us citances of test set and training set are equal.

Table 13 shows the P values of Mann–Whitney test for each of the difficulty measures.

The results of one-sided Mann–Whitney test with 95% confidence interval on the difficulty measures on Table 13 show that the null hypothesis is rejected which means that the SMOG index on unsolved-by-us citances of training set is statistically significantly lower than that of test set, which implies the more difficult textual content in test set that can lead to lower performance of the methods on test set rather than training set. To have a more robust conclusion, SMOG index is investigated separately in the next experiment. In this experiment, SMOG index is computed for all citances and their associated reference texts of each set. Then, Mann–Whitney test is computed on both sets of values to figure out if it shows that test set has more difficult text than that of the training set. In this experiment, the null hypothesis is that the distribution of SMOG index on all citances (and all reference texts) of training set and test set is the same.

As shown in Table 14 Mann–Whitney test with 95% confidence interval on SMOG values of both citances and reference texts show that SMOG values on the training set are statistically significantly lower than the test set which means the null hypothesis is rejected.

In the next experiment, we compute the absolute differences in SMOG values between citances (c) and their reference spans (SR(c)) i.e., $|\text{SMOG}(c) - \text{SMOG}(\text{SR}(c))|$ for the training set versus the same differences for the test set. Then we compute the Mann–Whitney test on both sets of values to figure out if the difference values of training set is

Table 11 Flesch–Kincaid grade level, SMOG readability index and Gunning's FOG index for citances and reference texts of Training documents of CL-SciSumm'17

TrainSetIDs	FK_{gr} Citances	FK_{gr} Ref. spans	$SMOG_{ind}$ Citances	$SMOG_{ind}$ Ref. spans	FOG_{ind} Citances	FOG_{ind} Ref. spans
C00-2123	10.3	10.3	8.8	8.8	10.4	10.4
C02-1025	9.9	11.5	8.8	8.8	10	10.4
C04-1089	7.2	12.7	8.8	11.2	8.4	11.6
C08-1098	13.0	11.1	11.2	11.2	13.2	10
C10-1045	11.5	7.2	8.8	8.8	11.6	7.2
C90-2039	13.8	10.7	11.2	8.8	10.4	8.4
C94-2154	11.1	12.7	8.8	8.8	10	11.6
D10-1083	7.6	11.9	8.8	11.2	7.6	9.6
E03-1020	13.0	9.9	8.8	8.8	13.2	11.2
E09-2008	15.0	16.6	11.2	11.2	14	12
H05-1115	11.5	13.1	11.2	8.8	8	9.6
H89-2014	11.5	11.1	11.2	11.2	11.6	8.8
I05-5011	10.3	10.7	8.8	8.8	8	9.6
J00-3003	8.0	15.8	8.8	13	6.8	11.2
J96-3004	8.7	13.0	8.8	11.2	8.8	12
J98-2005	12.7	9.5	11.2	11.2	11.6	8.4
N01-1011	11.9	12.3	8.8	8.8	10.8	11.2
N04-1038	12.3	14.2	11.2	11.2	10	9.6
N06-2049	8.4	13.1	8.8	11.2	8.4	10.8
P05-1004	8.7	14.6	8.8	11.2	8.8	11.2
P05-1053	8.0	14.2	3.1	11.2	8	10.8
P06-2124	11.5	11.9	8.8	11.2	10.4	8.4
P98-1046	10.3	15.0	8.8	8.8	10.4	12.8
P98-1081	8.0	9.1	8.8	8.8	8	8
P98-2143	12.3	15.8	11.2	13	10	12.4
W03-0410	12.3	13.0	11.2	11.2	11.2	12
W04-0213	11.5	13.8	11.2	11.2	9.2	10.4
W08-2222	13.8	15.0	11.2	13	11.6	11.6
W95-0104	8.7	9.5	8.8	3.1	10	8.4
X96-1048	12.3	9.9	11.2	8.8	11.2	8.8
Mean	10.83	12.30	9.57	10.15	10.05	10.28
Median	11.50	12.50	8.80	11.20	10.00	10.40
Std. Dev.	2.10	2.28	1.70	1.95	1.77	1.48

significantly less than their values of test set. The null hypothesis in this experiment is that the distribution of the difference in SMOG values is the same for the training and test set.

As shown in Table 15 the null hypothesis is rejected which means that the difference of SMOG values of citances and their reference spans in training set are significantly lower that the same value in test set. This comparison between the relative difference of citances as queries and reference spans as documents in the test set and training set can help us in explaining the lower results of reference span identification methods on test set.

Table 12 Flesch–Kincaid grade level, SMOG readability index and Gunning's FOG index for citances and reference texts of test documents of CL-SciSumm'17

TestSetIDs	FK_{gr} Citances	FK_{gr} Ref. spans	$SMOG_{ind}$ Citances	$SMOG_{ind}$ Ref. spans	FOG_{ind} Citances	FOG_{ind} Ref. spans
C98-1097	11.9	12.3	8.8	11.2	10.8	7.6
D09-1023	12.3	12.3	8.8	11.2	12.4	12.4
D10-1058	11.9	10.7	11.2	8.8	12	10.8
N09-1001	11.1	11.1	11.2	11.2	10	8.8
N09-1025	6.4	5.2	3.1	3.1	7.6	6.4
P00-1025	14.2	17.8	13	14.6	10.8	13.2
P07-1040	6.8	11.5	3.1	8.8	6.8	10.4
W06-3909	11.5	13.5	11.2	8.8	10.4	10
W09-0621	5.6	10.7	3.1	8.8	5.6	9.6
W11-0815	9.9	14.6	8.8	11.2	8.8	12.4
Mean	10.16	11.96	8.23	9.77	9.52	10.16
Median	11.30	11.86	8.8	10	10.2	10.2
SD	2.90	3.22	3.78	2.97	2.25	2.17

Table 13 *P* values of the Mann–Whitney test on Flesch–Kincaid grade level, SMOG readability index and Gunning's FOG index of unsolved-by-us citances of the test set and the training set

MW test (one-tailed)	FK_{gr}	$SMOG_{ind}$	FOG_{ind}
P value	0.751	0.042	0.760

Table 14 *P* values of the Mann–Whitney test on SMOG readability index of all citances and reference texts of the test set and the training set

MW test (one-tailed)	$SMOG_{ind}$ Citances	$SMOG_{ind}$ Reference texts
P value	0.045	0.000085

Table 15 *P* value for One-tailed MW test on $|\mathrm{SMOG}(c) - \mathrm{SMOG}(SR(c))|$ of training and test sets

MW test (one-tailed)	$SMOG_{ind}$ Abs. diff.
P value	0.043

Misclassifications

We looked at the misclassifications across each dataset for the various systems we tested. For certain citances most systems found at least one of the reference sentences chosen by annotators; for others, none were found. We used this information as a proxy for the difficulty of a citance. We show the distribution of "difficulty" across the training and test datasets in Fig. 5.

To generate this data, we calculated how many systems had found a correct reference span for each citance. For the test set, 46% of citances have at least one system correctly

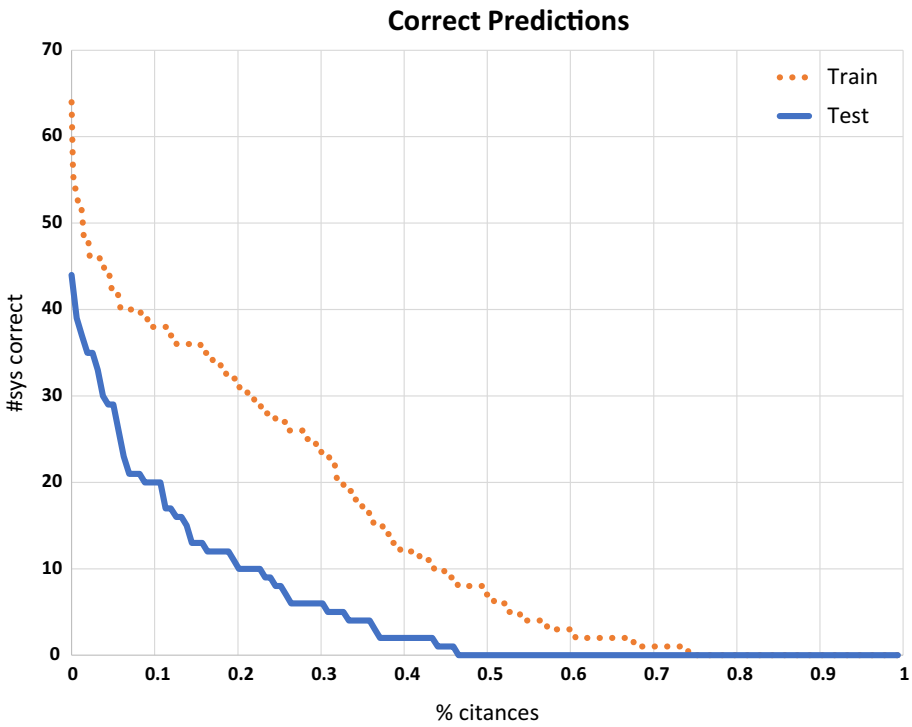

Fig. 5 The distribution of correct predictions among the citances of the training and test set

identify a chosen reference span. For the training set, the equivalent set of citances represents 74% of the total.

Using this misclassification information, we can generate new metrics. For instance, the metric $C_{D,\text{impos}}$ is the count of citances from document D that were unsolved by any of our systems. Another metric we use is $C_{D,\text{easy}}$ which refers to the count of citances from document D that were solved by at least half of our systems.

Correlations with unsolved-by-us citances

We calculate the two-tailed and one-tailed Mann Whitney (MW) tests between the number of unsolved-by-us citances $C_{D,\text{impos}}$ in the test set and the training set of reference documents. The metric is also calculated for the normalized ratio of unsolved-by-us citances to total number of citances in the two sets. We report the P values of the tests in Table 16.

Table 16 P value for two-tailed and one-tailed MW test on $C_{D,\text{impos}}$ count and ratio between training and test sets

MW test (one-tailed)	$C_{D,\text{impos}}$ Ratio	$C_{D,\text{impos}}$ Count
P value	0.0001	0.0494

Table 17 Spearman correlations across all reference documents in both training and test sets

Unsolved. Citance	Vocab. Size	Vocab. Ratio	Non-ASCII Words	Non-ASCII Chars.	$SMOG_{ind}$ Citances	$SMOG_{ind}$ Ref. spans
Count	0.0189	− 0.145	0.323	0.339	− 0.235	− 0.059
Ratio	− 0.1033	− 0.1056	0.0814	0.0809	− 0.153	− 0.086

We observe from the Mann–Whitney test results in Table 16 that the ratios of unsolved-by-us citances, $C_{D,\text{impos}}$, to total number of citances in D are more statistically significant in comparison to the raw count of the unsolved-by-us citances. We look at a distribution of Spearman's correlation between the set of unsolved-by-us citances: $C_{D,\text{impos}}$. The correlation values are calculated using an online software (Wessa 2017). We compare several parameters of the reference documents in both the Training and the Test sets with a raw frequency count of $C_{D,\text{impos}}$ and a normalized ratio of the same to total number of citances. In Table 17, we report the Spearman's correlation values for $C_{D,\text{impos}}$ with the Vocabulary[12] as well as Vocabulary ratio[13] of the document, number of Non-ASCII words and characters in the Reference document as well as the SMOG indexes in citances and reference texts for the test and training data.

Unsolved-by-us citances tend to be positively correlated with the frequency of Non-ASCII characters as well as Non-ASCII words in the text, which indicates encoding errors are a significant obstacle to proper retrieval. In addition we see some small inverse correlations for vocabulary ratio; the more difficult documents have a less diverse vocabulary for their size. We expect the effect to be larger than reported since we only compared with the vocabulary for an entire document; we expect the influence to be localized around citances and reference spans. Furthermore, the number of unsolved-by-us citances are negatively correlated with the SMOG index values, specially for the citances. It suggests that higher text difficulty can lead to lower number and ratio of unsolved-by-us citances (higher performance). The reason could be that higher text difficulty means more idiosyncratic word choices, which are easier to match. Lower text difficulty means vocabulary is simple and somewhat similar throughout, which makes matching difficult. A more detailed study of correlations between citances and reference sentences would help provide a better explanation.

Similarity with unsolved-by-us citances

We study the Jaccard similarity (JS) of the *unsolved-by-us*[14] citances, $C_{D,\text{impos}}$, for a reference document D, with the reference sentences (*RSs*) in the document. The experiments are repeated for the documents in the training and test sets separately. We use the following notations:

- *Unsolved-by-us Citances* ($C_{D,\text{impos}}$): Citances of document D that defeated all our proposed systems.

[12] Total number of unique words in the document.

[13] Ratio of vocabulary size to the total number of words in the document.

[14] Citances for which none of the retrieved sentences were relevant across all our proposed systems.

Table 18 P values of the two-tailed Mann–Whitney test on JS values of $C_{D,\text{impos}}$ and all RSs versus $C_{D,\text{easy}}$ and all RSs for each set separately

	Training set	Test set
MW test P value	$1.73e^{-45}$	0.0265

Table 19 P values of the two-tailed Mann–Whitney test on JS values of $C_{D,\text{impos}}$ and SR(c) versus $C_{D,\text{impos}}$ and $RS_{c,irrel}$ for each set separately

	Training set	Test set
MW test P value	$3.68e^{-10}$	$2.29e^{-18}$

- *Easy Citances* ($C_{D,\text{easy}}$): Citances of document D for which the reference spans (SR(c)) were correctly recognized by at least half of our proposed systems (if the number of correct predictions is $\geq$ 33, since we have a total of 66 systems)
- *Irrelevant Reference texts* ($RS_{c,irrel}$): This consists of all the reference sentences of D that are not in SR(c), i.e. $D - SR(c)$.

We make the following comparisons:

- Calculation of JS between: (a) $C_{D,\text{impos}}$ with all sentences of D and (b) $C_{D,\text{easy}}$ with all sentences of D.
- Calculation of JS between: (a) $C_{D,\text{impos}}$ and all the sentences in SR(c), and (b) $C_{D,\text{impos}}$ with all the irrelevant reference sentences, $RS_{c,irrel}$.

As explained earlier, the Mann–Whitney Non-parametric Test (Mann and Whitney 1947) is used to check the presence of statistically significant difference between two distributions without any constraints of normal distribution on the data. In this section, we calculate the two-sided Mann–Whitney test with 95% confidence interval for the Jaccard similarity (JS) value based distributions.

Table 18 gives the P value for the comparison between similarity distributions of the unsolved-by-us and easy citances in the train and test data with respect to the RSs in the documents. Here the null hypothesis is that the JS value distributions of the unsolved-by-us and easy citances with the corresponding reference sentences in the training and test documents are equal. In Table 19, we provide the same for the similarity values between $C_{D,\text{impos}}$ with the sentences in SR(c) as well as with the irrelevant sentences given a document. Similarly, in this experiment, the null hypothesis is that there is no statistically significant difference between the similarity distributions with respect to the reference spans and the irrelevant sentences in the training and test data. In both the cases, the P values prove that there is a statistically significant difference between the corresponding distributions for both test and training data. Thus the null hypothesis is rejected.

For our similarity measurements, we perform two necessary pre-processing steps on the data: removal of English stop words (Python NLTK library) and stemming (Python NLTK PorterStemmer).

Unsolved-by-us versus easy citances

We calculate the Jaccard similarity (JS) between the set of $C_{D,\text{impos}}$ with all the sentences of the reference document D (RSs). We also measure the JS values between $C_{D,\text{easy}}$ with RSs for the corresponding document. Figures 6 and 7 demonstrate the variation in the JS values between the unsolved-by-us citances and the easy citances respectively with the sentences for each reference document in training set, and Figs. 8 and 9 show the same but for the test set. In all the figures, we plot the Jaccard Similarity Values on the y-axis and the document names in the x-axis. For ease of demonstration in Figs. 6 and 7, we sort the training documents from 1 to 30 lexicographically and refer the readers to Table 20 for the filenames.

We observe there are very few easy citances in the test set (present in only 4 out of 10 documents) as shown in Fig. 9. In stark contrast, there are numerous citances present in the test set that were unsolved by our systems (Fig. 8). The training set is more balanced: we observe from Fig. 7 that nearly all documents have some easy citance; the situation for unsolved citances is similar.

Groundtruth v/s irrelevant citances

We calculate the Jaccard similarity (JS) values between $C_{D,\text{impos}}$ with two different types of texts from the document D—the reference span sentences in SR(c) for each c, and the irrelevant sentences of D $RS_{c,irrel}$. Figures 10 and 11 demonstrate the variation in JS values of $C_{D,\text{impos}}$ with the ground truth reference sentences and the irrelevant reference sentences

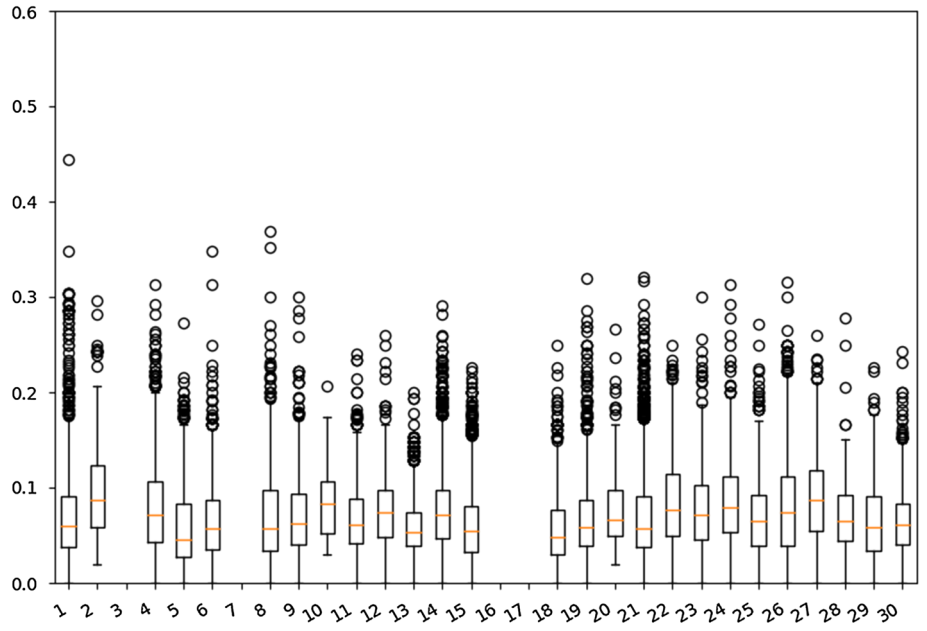

Fig. 6 JS values of unsolved-by-us citances with all the reference sentences in training set reference documents

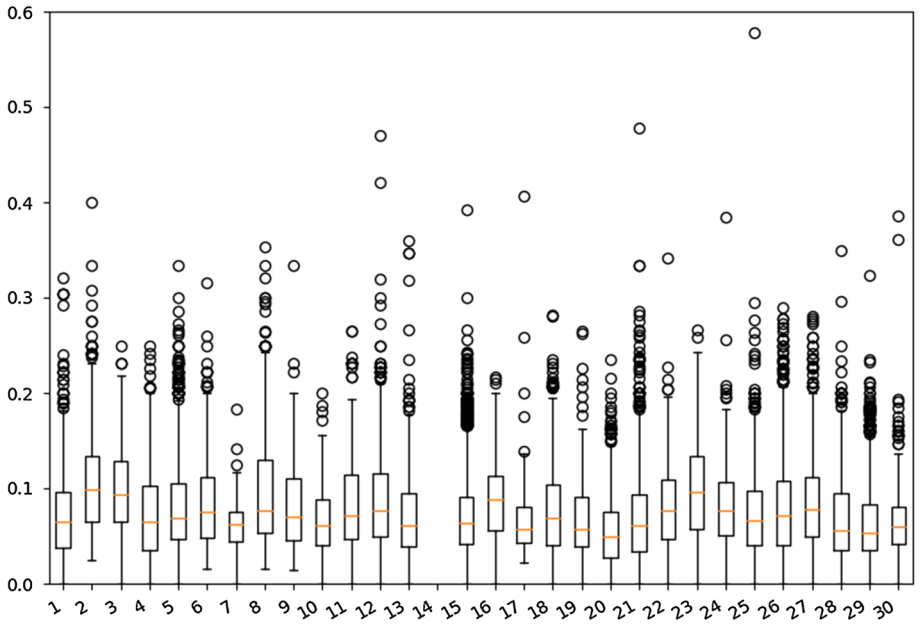

Fig. 7 JS values of easy citances with all reference sentences in training set reference documents

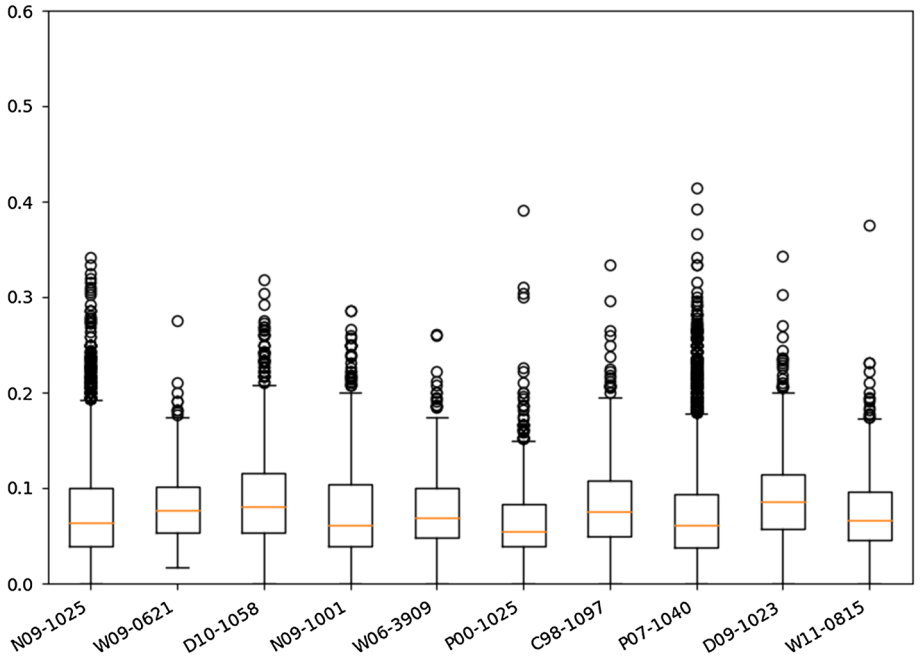

Fig. 8 JS values of unsolved-by-us citances with all reference sentences in test set reference documents

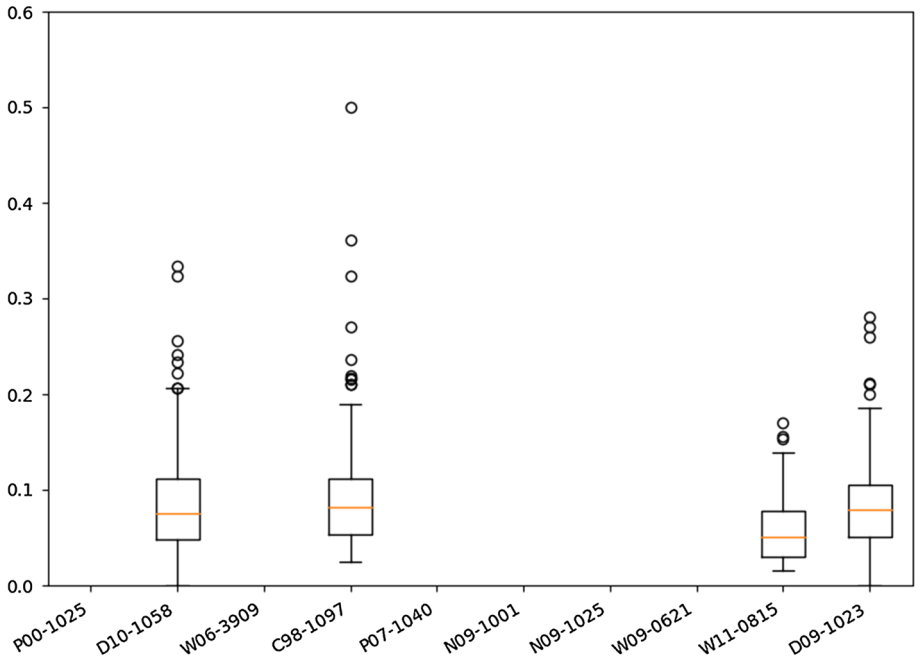

Fig. 9 JS values of easy citances with all reference sentences in test set reference documents

Table 20 Labels for the training set documents

ID	Name	ID	Name	ID	Name
1	C00-2123	11	H05-1115	21	P05-1053
2	C02-1025	12	H89-2014	22	P06-2124
3	C04-1089	13	I05-5011	23	P98-1046
4	C08-1098	14	J00-3003	24	P98-1081
5	C10-1045	15	J96-3004	25	P98-2143
6	C90-2039	16	J98-2005	26	W03-0410
7	C94-2154	17	N01-1011	27	W04-0213
8	D10-1083	18	N04-1038	28	W08-2222
9	E03-1020	19	N06-2049	29	W95-0104
10	E09-2008	20	P05-1004	30	X96-1048

respectively for the training data using box plots. Figures 12 and 13 show the same for the test dataset.

We observe that the Jaccard Similarity (JS) measures may not be a good measure of choosing the best set of reference sentences for a given citance. Comparing Fig. 10 with Fig. 11, we see higher JS values between the $C_{D,\text{impos}}$ with the 'irrelevant' reference sentences when compared to the reference span sentences for a large number of documents [for example, C00-2123 (Doc 1), D10-1083 (Doc 8)]. Better measures of "similarity" are needed to understand the relations between reference spans and citances.

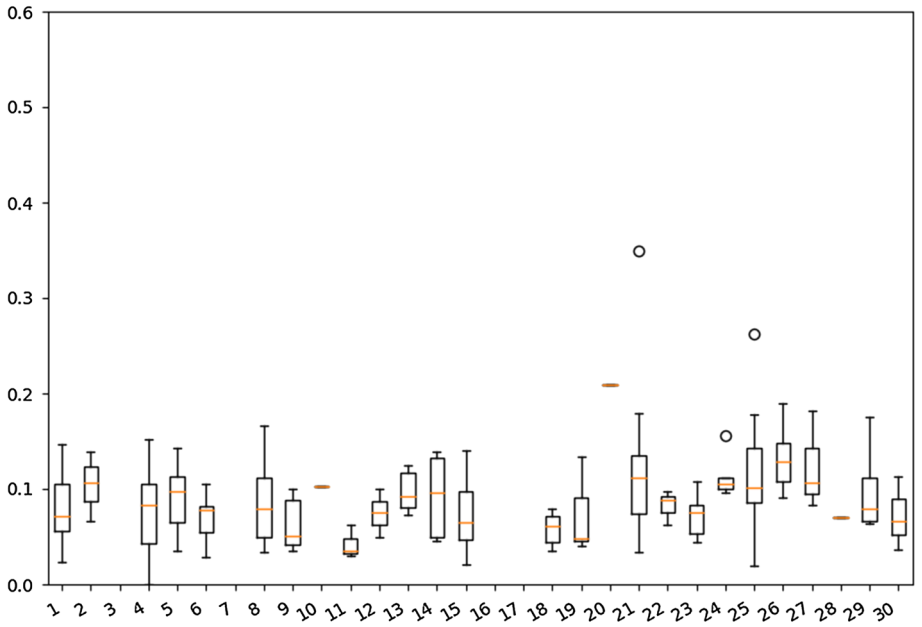

Fig. 10 JS values of unsolved-by-us citances with the reference span sentences (SR(c)) in training set reference documents

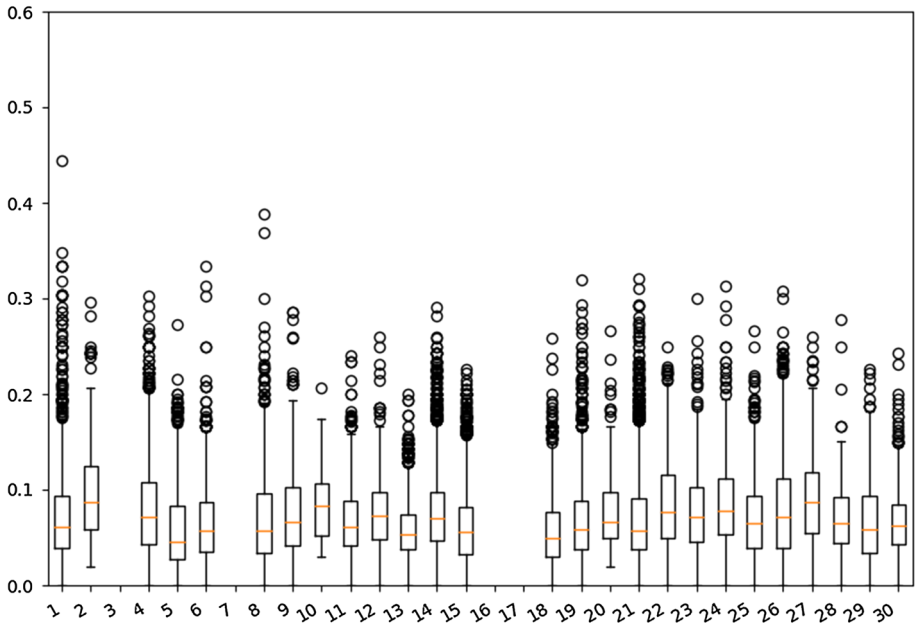

Fig. 11 JS values of unsolved-by-us citances with all irrelevant sentences in training set reference documents

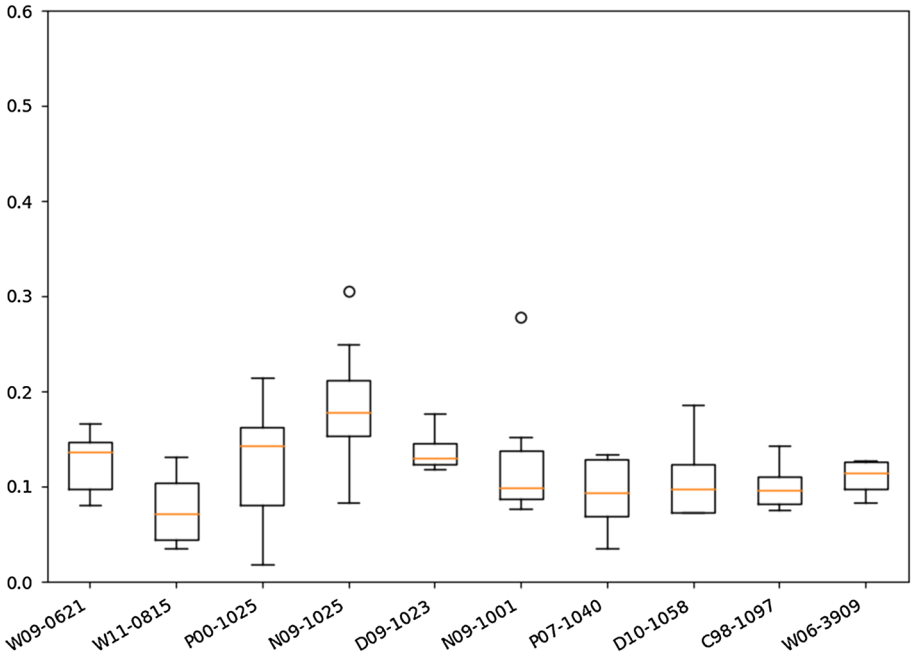

Fig. 12 JS values of unsolved-by-us citances with the reference span sentences (SR(c)s) in test set reference documents

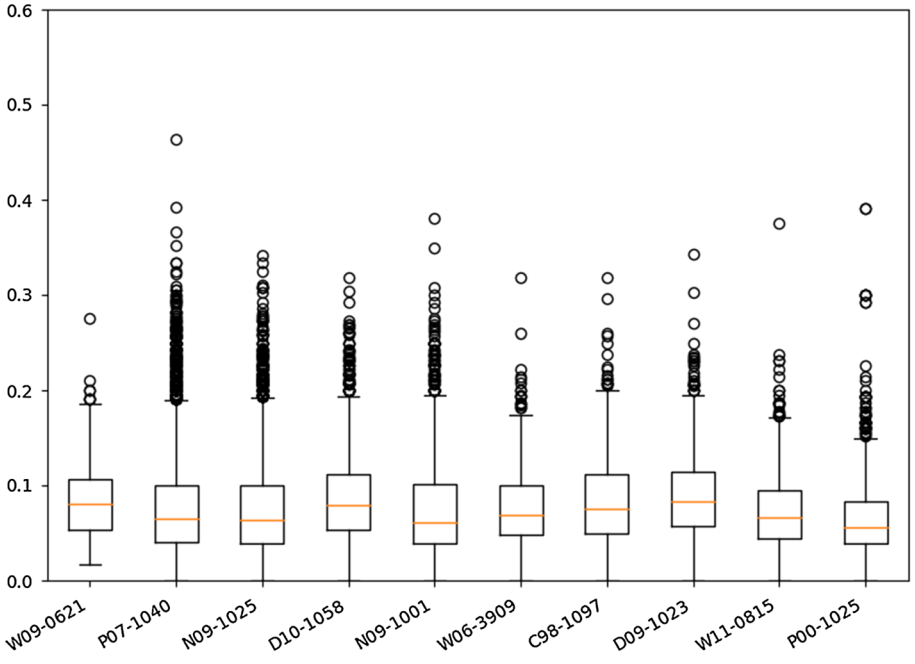

Fig. 13 JS values of unsolved-by-us citances with all irrelevant sentences in test set reference documents

Discussion

The distribution of discourse facets for the training and test sets is clearly quite different. At this time, we are unable to determine if this is because of a difference in annotators, or drift in annotator choices, or because the discourse facets are really different for the two sets.[15] Of course, it is also possible that this difference is due to some combination of all these reasons.

Our investigation into the differences between the test and training set also revealed a significant difference in the number of unsolved citances. Once we track the correlation of other variables with these "difficulty indicators" we observe that difficult documents tend to have a smaller vocabulary and tend to be easier to read. We initially expected difficult citances to be more complex, yet the opposite is true. This makes sense once we realize that a more varied vocabulary means it is easier to distinguish between sentences. Furthermore, a lower reading difficulty implies the use of more common words. A common word tend to be more general, which means it require greater contextual awareness since it can be used in many contexts. These results explain the effectiveness of TFIDF in the past and its degradation in the test set.

Conclusion

In this paper, we have presented several approaches and their performance on the three tasks of the CL-SciSumm 2017 shared challenge. We have also analyzed several interesting parameters of the 2017 training and test sets. For example, we found a significant difference in the facet distribution, and also differences in the readability levels. Our research suggests a tantalizing pattern: people tend to use "easier" (less technical) language when they refer to other papers and this makes it harder to identify reference spans. This was more frequently observed in the test set and we do see the performance of all our Task 1A methods declining on the test set.

References

AbuRaed, A., Chiruzzo, L., Saggion, H., Accuosto, P., & Bravo Serrano, À. (2017). Lastus/taln @ clsci-summ-17: Cross-document sentence matching and scientific text summarization systems. In *Proceedings of the Computational Linguistics Scientific Summarization Shared Task (CL-SciSumm 2017) organized as a part of the 2nd Joint Workshop on Bibliometricenhanced Information Retrieval and Natural Language Processing for Digital Libraries (BIRNDL 2017) and co-located with the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2017), August 11, 2017, Tokyo, Japan* (pp. 55–66).

Barrera, A., & Verma, R. (2012). Combining syntax and semantics for automatic extractive single-document summarization. In *CICLING* (Vol. LNCS 7182, pp. 366–377).

Bishop, C. M. (1995). *Neural networks for pattern recognition*. Oxford: Oxford University Press.

Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent dirichlet allocation. *Journal of Machine Learning Research*, *3*(Jan), 993–1022.

Blitzer, J., McDonald, R. T., & Pereira, F. (2006). Domain adaptation with structural correspondence learning. In *EMNLP 2007, Proceedings of the 2006 conference on empirical methods in natural language processing, 22–23 July 2006* (pp. 120–128). Sydney.

Bowman, S. R., Angeli, G., Potts, C., & Manning, C. D. (2015). A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 conference on empirical methods in natural language processing (EMNLP)*. Association for Computational Linguistics.

[15] The annotator information for the 2017 test set is not yet available.

Breiman, L. (2001). Random forests. *Machine Learning*, *45*(1), 5–32. https://doi.org/10.1023/A:1010933404324.

Burges, C. J. (2010). From ranknet to lambdarank to lambdamart: An overview. *Learning*, *11*, 23–581.

Cao, Z., Li, W., & Wu, D. (June 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 2016)* (pp. 132–138). Newark, NJ.

Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine Learning*, *20*(3), 273–297.

Dong, Y., Tian, R., & Miyao, Y. (2014). Encoding generalized quantifiers in dependency-based compositional semantics. In *PACLIC*.

Dubay, W. H. (2004). *The principles of readability*. Costa Mesa, CA.

Elkiss, A., Shen, S., Fader, A., Erkan, G., States, D., & Radev, D. (2008). Blind men and elephants: What do citation summaries tell us about a research article? *Journal of the American Society for Information Science and Technology*, *59*(1), 51–62. https://doi.org/10.1002/asi.20707.

Felber, T., & Kern, R. (August 2017). Graz university of technology at CL-SciSumm 2017: Query generation strategies. In *Proceedings of the 2nd joint workshop on bibliometric-enhanced information retrieval and natural language processing for digital libraries (BIRNDL 2017)*. Tokyo.

Fleiss, J. L. (1971). Measuring nominal scale agreement among many raters. *Psychological Bulletin*, *76*(5), 378–382. https://doi.org/10.1037/h0031619.

Freund, Y., & Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, *55*(1), 119–139.

Gambhir, M., & Gupta, V. (2017). Recent automatic text summarization techniques: A survey. *Artificial Intelligence Review*, *47*(1), 1–66.

Hoffman, M. D., Blei, D. M., & Bach, F. R. (2010). Online learning for latent Dirichlet allocation. In *Advances in neural information processing systems 23: 24th annual conference on neural information processing systems 2010. Proceedings of a meeting held 6–9 December 2010* (pp. 856–864). Vancouver, BC.

Jaidka, K., Chandrasekaran, M. K., Jain, D., & Kan, M. Y. (2017). Overview of the CL-SciSumm 2017 shared task. In *Proceedings of joint workshop on bibliometric-enhanced information retrieval and NLP for digital libraries (BIRNDL 2017)*. Tokyo: CEUR.

Jones, E., Oliphant, T., & Peterson, P., et al. (2001). *SciPy: Open source scientific tools for Python*. http://www.scipy.org/.

Karimi, S., Moraes, L., Das, A., & Verma, R. (August 2017). University of Houston@ CL-SciSumm 2017: Positional language models, structural correspondence learning and textual entailment. In *Proceedings of the 2nd joint workshop on bibliometric-enhanced information retrieval and natural language processing for digital libraries (BIRNDL 2017)*. Tokyo.

Kincaid, J. P., Fishburne, R. P., Rogers, R. L., & Chissom, B. S. (1975). *Derivation of new readability formulas (automated readability index, fog count, and flesch reading ease formula) for navy enlisted personnel*. Research Branch Report 875, Chief of Naval Technical Training: Naval Air Station Memphis.

Kullback, S., & Leibler, R. A. (1951). On information and sufficiency. *The Annals of Mathematical Statistics*, *22*(1), 79–86.

Kusner, M. J., Sun, Y., Kolkin, N. I., Weinberger, K. Q., et al. (2015). From word embeddings to document distances. *ICML*, *15*, 957–966.

Lauscher, A., Glavaš, G., & Eckert, K. (August 2017). University of mannheim@ CL-SciSumm-17: Citation-based summarization of scientific articles using semantic textual similarity. In Proceedings of the 2nd joint workshop on bibliometric-enhanced information retrieval and natural language processing for digital libraries (BIRNDL 2017). Tokyo.

Li, L., Zhang, Y., Mao, L., Chi, J., Chen, M., & Huang, Z. (August 2017) Cist@ CL-SciSumm-17: Multiple features based citation linkage, classification and summarization. In Proceedings of the 2nd joint workshop on bibliometric-enhanced information retrieval and natural language processing for digital libraries (BIRNDL 2017). Tokyo.

Lin, C. Y. (2004). ROUGE: A package for automatic evaluation of summaries. https://www.microsoft.com/en-us/research/publication/rouge-a-package-for-automatic-evaluation-of-summaries/.

Lu, K., Mao, J., Li, G., & Xu, J. (2016). Recognizing reference spans and classifying their discourse facets. In *Proceedings of the joint workshop on bibliometric-enhanced information retrieval and natural language processing for digital libraries (BIRNDL 2016)*.

Lv, Y., & Zhai, C. (2009). Positional language models for information retrieval. In *Proceedings of the 32nd international ACM SIGIR conference on research and development in information retrieval SIGIR 09*.

Ma, S., Xu, J., Wang, J., & Zhang, C. (August 2017). Njust@ CL-SciSumm-17. In *Proceedings of the 2nd joint workshop on bibliometric-enhanced information retrieval and natural language processing for digital libraries (BIRNDL 2017)*. Tokyo.

Mann, H. B., & Whitney, D. R. (1947). On a test of whether one of two random variables is stochastically larger than the other. *Annals of Mathematical Statistics*, *18*(1), 50–60. https://doi.org/10.1214/aoms/1177730491.

Manning, C. D., Raghavan, P., & Schtze, H. (2008). *Introduction to information retrieval*. Cambridge: Cambridge University Press. https://doi.org/10.1017/CBO9780511809071.

McLaughlin, G. H. (1969). SMOG grading—A new readability formula. *Journal of Reading*, *12*(8), 639–646. http://www.jstor.org/stable/40011226.

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. CoRR. arXiv:1301.3781.

Miller, G. A. (1995). Wordnet: A lexical database for english. *Communications of the ACM*, *38*(11), 39–41.

Mohammad, S., Dorr, B., Egan, M., Hassan, A., Muthukrishan, P., Qazvinian, V., et al. (2009). Using citations to generate surveys of scientific paradigms. In *Proceedings of human language technologies: The 2009 annual conference of the North American chapter of the association for computational linguistics* (pp. 584–592). Association for Computational Linguistics.

Moraes, L., Baki, S., Verma, R., & Lee, D. (2017). Identifying reference spans: Topic modeling and word embeddings help IR. *International Journal on Digital Libraries*. https://doi.org/10.1007/s00799-017-0220-z.

Moraes, L. F. T., Baki, S., Verma, R. M., & Lee, D. (2016). University of Houston at CL-SciSumm 2016: Svms with tree kernels and sentence similarity. 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)* (pp. 113–121). Newark, NJ, June 23, 2016. http://ceur-ws.org/Vol-1610/paper13.pdf.

Nakov, P. I., Schwartz, A. S., & Hearst, M. (2004). Citances: Citation sentences for semantic analysis of bioscience text. *Proceedings of the SIGIR*, *4*, 81–88.

Nanba, H., Kando, N., & Okumura, M. (2000). Classification of research papers using citation links and citation types: Towards automatic review article generation. *Advances in Classification Research Online*, *11*(1), 117–134.

Pontius, R., & Millones, M. (2011). Death to kappa: Birth of quantity disagreement and allocation disagreement for accuracy assessment. *International Journal of Remote Sensing*, *32*, 4407–4429.

Pramanick, A., Mandi, S., Dey, M., & Das, D. (August 2017). Employing word vectors for identifying, classifying and summarizing scientific documents. In *Proceedings of the 2nd joint workshop on bibliometric-enhanced information retrieval and natural language processing for digital libraries (BIRNDL 2017)*. Tokyo.

Prasad, A. (August 2017). Wing-nus at CL-SciSumm 2017: Learning from syntactic and semantic similarity for citation contextualization. In *Proceedings of the 2nd joint workshop on bibliometric-enhanced information retrieval and natural language processing for digital libraries (BIRNDL 2017)*. Tokyo.

Qazvinian, V., Radev, D. R., Mohammad, S., Dorr, B. J., Zajic, D. M., Whidby, M., et al. (2013). Generating extractive summaries of scientific paradigms. *Journal of Artificial Intelligence Research (JAIR)*, *46*, 165–201.

Quinlan, J. R. (1986). Induction of decision trees. *Machine Learning*, *1*, 81–106.

Salton, G., & Buckley, C. (1988). Term-weighting approaches in automatic text retrieval. *Information Processing and Management*, *24*(5), 513–523.

Shivam Bansal, C. A. (2017). *Textstat 0.4.1*. https://pypi.python.org/pypi/textstat.

Tian, R., Miyao, Y., & Matsuzaki, T. (2014). Logical inference on dependency-based compositional semantics. In *Proceedings of the 52nd annual meeting of the association for computational linguistics*. ACL.

Verma, R. M., & Lee, D. (2017). Extractive summarization: Limits, compression, generalized model and heuristics. *Computacion y Sistemas*, *21*(4), 787–798.

Wessa, P. (2017). *Spearman rank correlation (v1.0.3) in free statistics software (v1.2.1) office for research development and education*. https://www.wessa.net/rwasp_spearman.wasp/.

Zhang, D., & Li, S. (August 2017). PKU@CL-SciSumm-17: Citation contextualization. In: *Proceedings of the 2nd joint workshop on bibliometric-enhanced information retrieval and natural language processing for digital libraries (BIRNDL 2017)*. Tokyo.

Zhao, K., Huang, L., & Ma, M. (2016). Textual entailment with structured attentions and composition. In *COLING 2016, 26th international conference on computational linguistics, proceedings of the conference: Technical papers, December 11–16, 2016, Osaka, Japan* (pp. 2248–2258). http://aclweb.org/anthology/C/C16/C16-1212.pdf.