IR&TM-NJUST @ CLSciSumm-19

Shutian Ma, Heng Zhang, Tianxiang Xu, Jin Xu, Shaohu Hu, Chengzhi Zhang*

Department of Information Management, Nanjing University of Science and Technology, Nanjing, China, 210094
mashutian0608@hotmail.com, 525696532@qq.com, 1832862727@qq.com, xujin@njust.edu.cn, 191226105@qq.com,zhangcz@njust.edu.cn

Abstract. This paper introduces IR&TM-NJUST system submitted in CL-SciSumm 2019 Shared Task at BIRNDL 2019 Workshop. Overall, there are three basic tasks. Task 1A is to identify cited text spans in reference paper. Briefly, we solve this problem by using multi-classifiers and integrate their results via voting system. Compared with our CLSciSumm-18 system, this year we make feature selection based on correlation analysis, apply similarity-based negative sampling strategy to build training dataset and add deep learning models for classifications. For task 1B, which is to identify facets of cited text, we firstly calculate the probability that each word would belong to the specific facet. Then, logistic regression models are trained using these probability features and character-based features. When predicting over test data, prior rules are added to obtain final result. As to Task 2, in order to obtain a logical summary, we apply two ways to organize sentences into groups following the logical order. The first method is to calculate their similarity with three abstract parts segmented in order. Second is to divide them into groups based on the recognized facet from task 1B. By ranking via different features, we pick out important sentences from each group and generate the summary in logical sequence within 250 words.

1 Introduction

As the most important communication media between researchers, number of scientific publications has increased rapidly from early on. In order to alleviate such paper overload, scientific summarization systems have been investigated and implemented for many years [1, 2], which are software tools and techniques providing a summary for the scientific paper to a user. Traditional models focused on aggregating all citances (citation sentences) that cite one unique paper for summarization [3, 4]. However, this will lead to problems when citances carry different viewpoints. Besides, detailed information can't be revealed from citances since these are already general comments from citing authors. Originating from TAC 2014 Biomedical Summarization Track, a series of Computational Linguistics Scientific Document Summarization Shared Task

* Corresponding Author.

(CL-SciSumm)[1] are proposed to generate summaries based on cited text spans (CTS). The new mechanism is based on reference paper itself, which can provide more reliable context information. There are two main steps in CL-SciSumm and the first step contains two sub-tasks. Below are the detailed descriptions.

Given: *A topic consisting of a Reference Paper (RP) and Citing Papers (CPs) that all contain citations to the RP. In each CP, the citances have been identified that pertain to a particular citation to the RP.*

Task 1A: *For each citance, identify the cited text span in the RP that most accurately reflect the citance. These are of the granularity of a sentence fragment, a full sentence, or several consecutive sentences (no more than 5).*

Task 1B: *For each cited text span, identify what facet of the paper it belongs to, from a predefined set of facets.*

Task 2: *Finally, generate a structured summary of the RP from the cited text spans of the RP. The length of the summary should not exceed 250 words.*

Our team has attended the task in 2017[5] and 2018[6]. This year we propose new strategies for all the three subtasks for CL-SciSumm 2019[7]. In task 1A, more efficient features are picked out, negative sampling is utilized to alleviate the imbalanced-data problem and neural network models are constructed additionally. For task 1B, we identify facet according to the probability of word learned from training set. In task 2, we firstly arrange sentences following logical order and then select important ones to generate summary.

2 Related Work

With more publications coming out, there is an urgent demand to build scientific summarization systems to help scholars quickly move into a new research field. Traditional approaches utilize citations which could be a good resource to understand the main contributions of a paper and how that paper affects others. Since citations might exist subjective opinions from authors, a new framework using cited text spans from reference paper for summaries is proposed. It can avoid the situations that citations hold different views from each other [8-10]. We will present related work about cited text span based summarizations in CL-SciSumm 2017 and 2018.

In order to solve task 1, many teams will do feature extraction firstly since task 1A and 1B can both be seen as a classification task. Basically, there are two main types of features which are widely used: similarity-based [11] and position-based features[12]. Referring to similarity-based features, researchers are making efforts to find linkages between citance and reference sentences. The first kind of linkage is constructed from character or chunk level such as using N-gram[12], Longest Common Subsequence, Word Mover's Distance[11] and so on. Meanwhile, sentence similarity are also obtained via traditional models like, TF-IDF, Jaccard, modified Jaccard, BM25[13]. In order to mine more semantic information, teams are also using word embeddings[13]

[1] This task is organized annually from 2016 to 2019. Website for CL-SciSumm 2019 is available at: http://wing.comp.nus.edu.sg/~cl-scisumm2019/

learned from corpus like ANN, ACL and Google News. Lexical resources such as Wordnet is also applied. Position-based features are normally the physical location or relative location of sentences in the paper. Such features contain location of paragraph, document position ratio, paragraph position ratio and so on[11]. Lots of features are created which seem to cover all possible ways to mine relations between two sentences in task 1A and hidden facet patterns in task 1B. Single classifier or voting system using multi-classifiers are adapted with those features. Popular models like Random Forest, Decision Tree, KNN, SVM XGBOOST[6] and neural network all have been used already, combining with different ensemble strategies at the same time[14]. However, the state-of-art performance are still remained to be improved. By exemplifying related work, there are several shortcomings, which can also be potential for optimizations.

First of all, there should be one more step after constructing all kinds of features, which is the feature selection. Feature selection is to select some of the most effective features from a group of features. The identification of relevant features in task 1 is an important step towards gaining insight underlying the data. Other advantages of feature selection include the ability of the classification system to obtain good or even better solutions using such a restricted subset of features [15, 16]. Studies have approved that the Naive Bayes can achieve considerably better results when feature selection is applied [17], yet also the SVM can benefit from feature selection [18]. Secondly, although some current systems have dealt with imbalanced data problems, there is still lots of room for improvement. To balance numbers between positive and negative samples in origin data set, most of teams choose the reference sentences randomly as negative samples [19]. Oversampling strategy has been utilized in several systems for task 1A, such as using SMOTE, ENN and NN technique to increase the number of positive samples or to decrease the number of negative samples [14, 20]. Referring to task 1B, system tend to set up more rules to classify facet based on lexical evidence[21] or use class weights[22]. Therefore, more strategies could be utilized to alleviate such imbalanced-data problem. As it is observed that researchers have applied some neural network models, such CNN[11, 23] and LSTM. For example, WING_NUS team use CNN and LSTM model to convert vocabulary indexed text and create a classification model and a ranking model separately[22]. University of Houston use LSTM units to learn the dependencies across the textual pairs [24]. So far, systems have taken few features as input for neural models and ensemble strategy haven't been applied yet.

When doing task 2, it usually contains two main steps: sentence ranking and sentence grouping[11]. Sentence ranking is the action that rank sentence based on several features and select those which are in the top. Then we could combine them in a certain order to generate final summary. LaSTUS/TALN team's system is a trainable sentence scoring, sentence ranking and sentence extraction algorithm which optimally combines the contribution of several numerical features to produce sentence scores[25]. NLP-NITMZ ranked the generated sentences from reference paper a score based on Jaccard similarity score between all the cited text and reference text. They also considered sentence length and location, where in summary there should be at least one sentence from introduction, implementation, methods and results[26]. Since task 2 is based on task 1, it would be more effective if we can improve task 1 performance greatly.

3 Methodology

3.1 Task 1A

We approach task 1A as the problem to verify which sentence in reference paper can reflect citances directly. This year, optimization is conducted from three aspects: feature selection, negative sampling and neural network models for classification.

Feature construction. To have more efficient features, we conduct correlation analysis over a new feature set on the basis of old features previously used [6]. Few features are added compared with previous work, such as longest common subsequence, longest common substring, WordNet similarity and so on. Description of new feature set are given in Table 1.

Table 1 New Feature Set for Correlation Analysis

Feature	Feature Description
Sentence length(*sl*)	The number of words in candidate CTS.
Sid(*sid*)	The serial number of candidate CTS in full text.
Ssid(*ssid*)	The serial number of candidate CTS in paragraph it belongs to.
Longest common subsequence(*lseq*)	See citance and candidate CTS as two sets of sequences with words as basic unit, find the longest subsequence (not necessarily consecutive in original sequences) common to two.
Longest common substring(*lstr*)	See citance and candidate CTS as two sets of strings with words as basic units, and find the longest string(s) that is a substring(s) (required to occupy consecutive positions within the original strings) of two.
Sentence position(*senp*)	The ratio of Sid and the number of sentences in full text.
Section position(*secp*)	The position of paragraph candidate CTS is located, divided by the number of paragraphs in full text.
Inner position(*innp*)	The ratio of CTS's Ssid and the number of sentences in the paragraph it belongs to.
TextSentenceRank(*tsr*)	The weight of candidate CTS modeled by TextRank[27] algorithm.
Dice similarity(*dice*)	Segment citance and candidate CTS into sets of words($\boldsymbol{s_1}, \boldsymbol{s_2}$). It is calculated by: $$\frac{2 * intersection(s_1, s_2)}{length(s_1) + length(s_2)}$$
Jaccard similarity(*jacc*)	Segment sentences into set of words, and calculate the division of the intersection and union between two sets.
Doc2Vec similarity(*d2v*)	Represent sentences as low-dimensional and dense vectors via Doc2Vec algorithm, and calculate cosine value between two vectors.
Levenshtein distance(*leven*)	Calculate the average of Levenshtein distance (the minimum number of single character edits required to change one to the other) for all the words between two sentences.
LDA similarity(*lda*)	Represent probability distribution of sentences according to their topics, and calculate cosine value between two sentence vectors.
WordNet similarity(*wn*)	Based on WordNet ontology, calculate the average of the similarity between words from two sentences.

Bigram_overlap(*bo*)	Segment sentences into sets of bigram, and calculate the number of overlap between two sets.
Word_overlap(*wo*)	Segment sentence into sets of words, and calculate the number of overlap between two sets.
Word2Vec similarity(*w2v*)	Represent words as low-dimensional and dense distributed representation by Word2Vec algorithm, and calculate the average of the similarity between words from two sentences via cosine value.

Then, we calculate Pearson Correlation Coefficient [28] between each feature and annotated label to find out efficient features. According to Table 2, we keep six features (bold) to be our final features which are with high correlation and significance.

Table 2. Pearson Correlation Coefficients between Different Features and Annotated Labels

sl	*sid*	*ssid*	***lseq***	***lstr***	*senp*	*secp*	*innp*	*tsr*
0.129**	-0.126**	-0.104**	**0.385****	**0.305****	-0.126**	-0.102**	-0.067**	0.092**
dice	***jacc***	*d2v*	*leven*	*lda*	*wn*	***bo***	***wo***	*w2v*
0.343**	**0.355****	0.050**	-0.071**	0.127**	0.066**	**0.327****	**0.374****	0.184**

**. Correlation is significant at the 0.01 level (2-tailed).

*. Correlation is significant at the 0.05 level (2-tailed).

Table 3 Combinations between Feature Set and 4 Classifier

Combination	Classifier	Feature Set
1	SVM(RBF)	*tf_idf_sim, idf_sim, sid*
	SVM(Linear)	*tf_idf_sim, idf_sim, bigram, lda*
	Decision Tree	*tf_idf_sim,idf_sim,jacc,senp,sid,innp,w2v*
	Logistic Regression	*tf_idf_sim,idf_sim,jacc,secp,lda*
2	SVM(RBF)/SVM(Linear)/Decision Tree/Logistic Regression	*lseq, lstr, dice, jacc, bo, wo*

For ensemble systems using machine learning classifiers, we utilize two set of features to feed them into each algorithm (Table 3). First combination is a continuation of previous work. Second one is the 6 features obtained from correlation analysis and they will be combined with all 4 classifiers. Except those have been mentioned in Table 1, there are three features from the old set and their descriptions is given in Table 4.

Table 4 Three Features from old Set for Classifiers

Feature	Feature Description
tf_idf_sim	Cosine value between two sentence vectors represented by TF-IDF
idf_sim	Adding up IDF values of the same words between two sentences
bigram	Bi-gram matching value, if there is any of bi-gram matched between two sentences, this value is 1; otherwise 0

Negative data sampling. When dealing with imbalanced data, current studies rely on changing proportion of positive and negative data by sampling randomly or adding/removing data. However, such methods assumed that all selected or created data is meaningful to imply the patterns over real data set. According to released data set of

CL-SciSumm 2017 in Figure 1, there are much more negative samples compared with positive ones. Therefore, a careful choice of negative training examples is critical for model performance. In this paper, we use three types of sentences in the reference paper that are not real cited text spans for each citance to build negative examples:

- Sentences that its similarity with corresponding citance is the highest.
- Sentences that its similarity with corresponding citance is the lowest.
- Sentences that its similarity with corresponding citance is in the middle of the highest and lowest value.

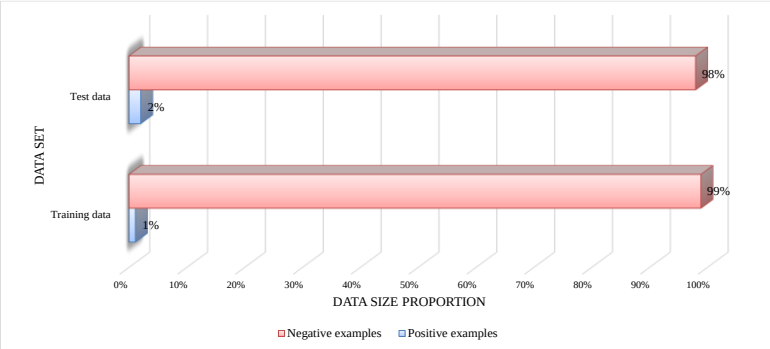

Figure 1 Imbalanced-data Problem in CL-SciSumm 2017

When calculating similarity between sentences, we use 8 different similarity metrics and evaluate them based on ten fold cross validation over training set. As it shown in Figure 2, *Bigram_overlap* and *Word_overlap* perform better than the other features, we then utilize these two approaches to do negative sampling and the obtained training set are then fed into the ensemble system with four classifiers using old feature set.

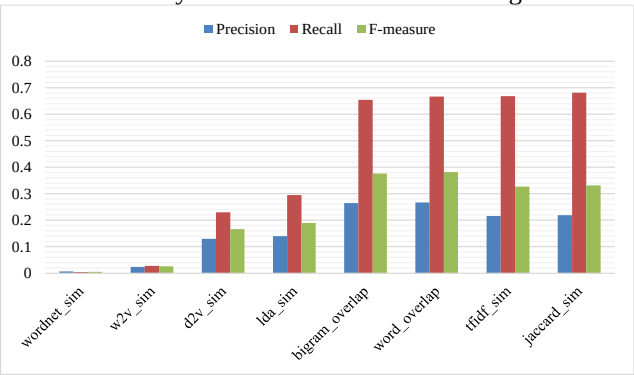

Figure 2 Ten Fold Cross Validation over Training Set of 8 Similarity Metrics

According to Table 5, we use *Bigram_overlap* feature to calculate similarity between sentences to conduct negative sampling. Besides, ratio of positive to negative examples is set to be 1:10. For each citance, we select 5 sentences whose similarity with corresponding citance is the highest, 3 sentences whose similarity with corresponding citance is the lowest, and 2 sentences whose similarity with corresponding citance is in the middle of the highest and lowest value.

Table 5 Cited Text Spans Identification using Two Negative Sampling Approaches

Negative sampling approach	Precision	Recall	F1
Bigram_overlap	0.2723	0.2316	*0.2503*
word_overlap	0.1875	0.1660	0.1750

Classifier models. Totally, there are four different systems we built for Task 1A: 4-classifers, 3-classifiers, Single MLP and Ensemble MLP. We firstly integrate several popular classifiers: SVM (RBF), SVM (Linear), decision tree and logistic regression. Two voting systems are then built: one is 4-classifiers containing all classifiers, another one is 3-classifiers where we remove one in each system.

For Combination 1 in Table 3, voting mechanism of multi-classifiers is the same with previous system [6]. Threshold of voting system is tested on 0.2, 0.4 and 0.6. For Combination 2 which applies new features, we set the equal weight for each classifiers. If there is only one classifier identifies the sentence to be cited text span, then the voting system value is 1. Threshold of voting system is tested on 1, 2, 3 and 4.

Except these two multi-classifiers, two MLP-based structures are also built for predicting the probability that the sentence would be cited text span: Single MLP and Ensemble MLP.

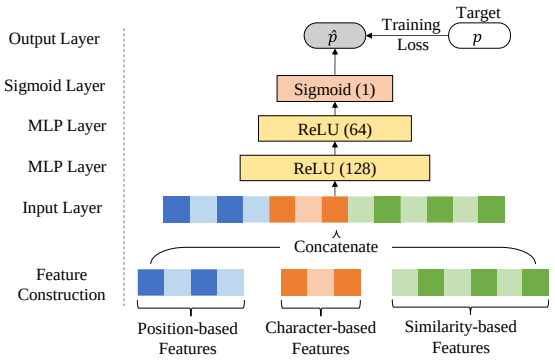

Figure 3 Illustration of Single MLP Framework

Given the citance and target sentence in reference paper to be identified, Single MLP uses 13 features consisting of the character-based, position-based and similarity-based features as input for classification. For character-based features, we take *sl* and *bigram* into considerations, for position-based features, we use *sid*, *ssid*, *senp*, *secp* and *innp*. Similarity-based features are *jacc*, *lda*, *wn*, *w2v_acl* and *w2v_google*. *w2v_acl* is obtained by word embedding trained by ACL corpus[2]. *w2v_google* is obtained by word embedding trained by Google news corpus[3]. Detailed descriptions of these features can be found in Table 1. We concatenate features to be input layer, and it's then followed with two separate and fully connected hidden layers. Activation function and neurons numbers are given in Figure 3. Probability of the target sentence to be cited text span $\hat{p}$ is finally achieved by applying a sigmoid function over the last layer. Binary cross

[2] Available at: http://acl-arc.comp.nus.edu.sg/

[3] Available at: https://github.com/mmihaltz/word2vec-GoogleNews-vectors

entropy is chosen to be loss function and we use Adam optimizer for training[29].

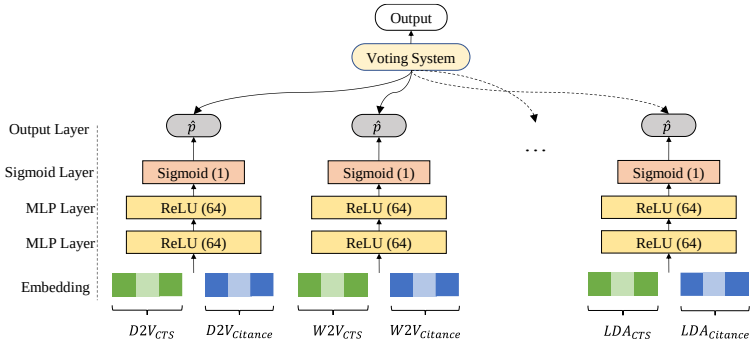

Figure 4 Illustration of Ensemble MLP Framework

Ensemble MLP model is constructed in the structure shown in Figure 4. Given the citance sentence and target sentence (CTS) in reference paper to be identified, we firstly learn their vector-based representations using Doc2Vec, LDA and Word2Vec. Two Word2Vec models are trained. *Word2Vec_ab* is obtained by word embedding based on abstract information of papers. *Word2Vec_body* is obtained by word embedding based on full text. And then each pair of citance and CTS vectors will be concatenated as input layer. Similar with Single MLP, after two fully connected hidden layer, probability of the target sentence to be cited text span is finally achieved by applying a sigmoid function. Then, we use a voting system to generate the final results. To find the proper parameter of optimizer and loss function, we trained 36 different model according to the combinations between the embedding, optimizer and loss function (Table 6). For the probability it predicts, we also set threshold for judging if the sentence is cited text span or not, parameter is set to be 0 to 1 and 0.001 as interval.

Table 6 Different Models for Embedding, Optimizer and Loss Function

Module	Model
Embedding	Doc2Vec, LDA, Word2Vec_ab, Word2Vec_body
Optimizer	Adam, RMSprop, SGD
Loss Function	Binary Cross Entropy, Categorical Cross Entropy, Mean Squared Error

After testing models over the test data set, Top 3 and Top 5 models are selected to be integrated together as the final ensemble systems. Referring to voting mechanism, if there are over 2 votes in Top-3 ensemble MLP model or over 3 votes in Top-5 ensemble MLP model, we will identify the sentence to be cited text spans.

3.2 Task 1B

In this task, we need to identify the facet of cited text spans and there are five facets to be chosen: *Hypothesis*, *Aim*, *Implication*, *Method* and *Result*. Basically, we apply logistic regression for this classification task combining with several prior experiences.

Feature construction. Firstly, text preprocessing is conducted to remove citation markers like "*King [8]*" or "*(blei et al., 2003)*". Then, we extract keywords and then a unique word list is obtained for each sentence. Two type of features are constructed here: character-based and probability-based. As it is observed from training set,

sentences belonging to *Result* facet would contain numerical symbols like percent or decimal point. Therefore, the first two character-based patterns are the symbol matching value. If there is any percent matched in the sentence, percent feature value is 1; otherwise 0; decimal point feature is in the same way. The second feature is a five-dimension vector. Each dimension is the probability that the sentence would belong to the specific facet. Following are the steps that how we generate this feature:

(1) Calculate document frequency (DF) of each word and filter the word whose DF value is not between 2 and 100.
(2) Calculate probability that each word belongs to five facets according to labeled facet in training data set.
(3) Sum up probability of words in one sentence for each facet, do normalization processing to obtain final probability that this sentence belongs to each facet.

For each pair of citance and cited text span, we conduct the third step. Then, for each facet, we add probability of citance and probability of cited text span in the proportion of 4:1 to get the final probability. This will also generate the five-dimension vectors, which is the feature of belonging probabilities.

Training and testing. In order to train logistic regression model, we need to generate training data for task 1B. Since there are much more *Method* facet in training data, for those sentences with two facets, if one of the two labels is *Method*, we will only keep the other label to be this sentence's facet. Otherwise, we will keep the first label to be the sentences' facet according to appearance order. Furthermore, we train two logistic regression model. Model 1 uses original training data obtained after allocation of facet labels. Model 2 uses the new data constructed by increasing data proportion of *Aim*, *Hypothesis* and *Implication* facet. We quintuple *Aim* and *Hypothesis* labeled sentences and triple *Implication* labeled sentences in Model 2. Three prior rules are also added:

Rule 1: Lexical matching rule, which is to use some pre-defined dictionaries of specific facet and match them with the identified cited text span without preprocessing. For *Aim* facet, pre-defined dictionary contains *adapted to*, *draws on*, *task of*, *procedure for*, *focus of*, *goal of*, *goal to*. For *Implication*, dictionary contains *believe*, *limitation* and *unrealistic*. For *Result*, dictionary contains *less than*, *lower than*, *showing*, *shows*, *show* and *shown*. Besides, if sentence can be matched with the word in *Result*, then this sentence is only identified as *Result*.

Rule 2: Probability threshold rule, which is to set thresholds for facet probabilities. If the identified probability of *Implication* facet is between 0.32 and 0.4, then this sentence will be only identified as *Implication*. If identified probability of *Implication* is between 0.2 and 0.21, then sentence will be identified as *Implication*. If identified probability of *Hypothesis* is more than 0.08, then sentence will be identified as *Aim* and *Hypothesis*. If identified probability of *Result* facet is more than 0.33, then sentence will be identified as *Result*. Finally, if there is no probability for all facets, then sentence will be identified as *Method*.

Rule 3: *Method* facet rule, which is to identify the sentence to be *Method* facet, if the above approaches can't classify it as any of the five facets.

For the final submissions, we conduct three combinations between logistic regression model and rules, which are $(Model\ 1, Rule\ 1, Rule\ 3)$, $(Model\ 2\ , Rule\ 1, Rule\ 3)$ and $(Model\ 2\ , Rule\ 1, Rule2, Rule\ 3)$.

3.3 Task 2

Summary generations can be divided into two steps in our system. First is to group sentences into different clusters. Second is using ranking features to extract sentence from each cluster and combine them into a summary. Totally, we utilize two different strategies for the submitted system.

The first strategy is based on the previous work [6]. Since abstract is a concise description, we assume that it will contain motivation, approach and conclusion. In order to generate a summary in logical order, we apply rule-based method based on writing styles. When people write summaries like abstract, they often start with some fixed phrases, such as "this paper", "in this paper" or "we". So, if the first sentence doesn't start with these phrases, it will be about motivation for most times. Similarly, the last sentence in abstract is usually about results or conclusions. Based on these rules, we split abstract into segments, each identified text span is selected into different groups based on their similarity with these segments. Here, we use linear sum of Jaccard, IDF and TF-IDF similarities. After grouping, we rank sentences within each group based on features of three similarities, sentence length and sentence position. Since importance of features vary from each other, we set different weights to show differences. For the three similarity-based features, they contain more semantic relations between identified text spans and abstract sentences. Therefore, we allocate the same weights to them which are bigger than sentence length and sentence position. Formula is shown below:

$$Score_i = 2.5S_{Jacc} + 2.5S_{IDF} + 2.5S_{TFIDF} + 1.25S_{Length} + 1.25S_{Position} \quad (1)$$

Finally, we choose the first sentence from each cluster based on the ranking score to build summary until the length of summary reaches 250 words.

The second strategy one is based on the identified facet obtained in task 1B. Similar with the previous strategy, we want to split identified cited text span into groups based on some logical evidences. Since we have recognized the facet, the second strategy will make use of these results. The sentences which carry *Aim* and *Hypothesis* facet will be grouped into the first group. The sentences which is *Method* will be in the second group. The left ones identified as *Result* and *Implication* will make up the third group. Then, we also rank the sentences within each group. Since keywords can represent more meaningful information, we extract keywords and calculate the Jaccard similarity with abstract and give scores for each sentence.

$$Score_i = S_{Jacc} \quad (2)$$

Then, we choose the first sentence from each group based on the ranking score to build summary until the length of summary reaches 250 words.

4 Experiments

4.1 Data and Tools

The training data set we use this year is made up by two corpora. The first dataset comprised 40 annotated sets of references and their citing papers from the open access

research papers in the computational linguistics domain[4]. The second one is a 1000 document set that were automatically annotated to be used as training data from SciSummNet [30], which is expanded from the CL-SciSumm project released by Yale LILY lab[5]. Since the auto-annotated data is available only for Task 1A. We use the auto-annotated data with the first data for training models for Task 1A and use the manually annotated training data from 40 document sets for Task1B.

When doing corpora processing, we remove the stop words and stem words to base forms by Porter Stemmer algorithm[6]. Then, we applied Word2Vec and Doc2Vec model in Genism[7] and python package of LDA[8] model to represent documents. All the classifiers were done via Scikit-learn python package[9]. Keywords are extracted via a python package rake_nltk[10]. Neural network models are built using Keras. Source code of our system will be made available at: https://github.com/michellemashutian/NJUST-at-CLSciSumm/tree/master/MZXXH_NJUST-at-2019.

4.2 Submission Results

Task 1A. Since the multi-classifiers strategy is conducted after careful feature selection, here we only report the results of MLP models and how we select trained model for final system. Firstly, for the single MLP, we run for 10 times and test the threshold of the predicted probability from 0.4 to 0.98, with 0.02 as interval. Threshold, average of precision, recall and F-measure are shown in Figure 5. As it is shown in Figure 5, with the increasing of threshold, F-measure is getting bigger since the precision is increased, however the recall is decreasing. For final submissions, we provide 5 different results trained by the single MLP model and threshold for judging the candidate sentence to be cited text span is 0.55, 0.65, 0.75, 0.85 and 0.95.

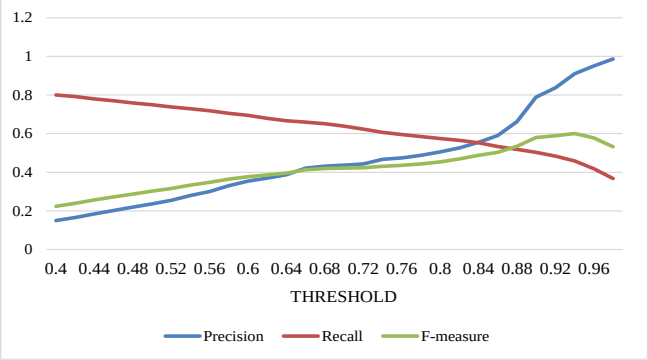

Figure 5 Average of Precision, Recall and F-measure of 10 runs for Single MLP

[4] Available at: https://github.com/WING-NUS/scisumm-corpus
[5] Available at: https://michiyasunaga.github.io/projects/scisumm_net/
[6] Available at: http://snowball.tartarus.org/algorithms/porter/stemmer.html
[7] Available at: https://radimrehurek.com/gensim/
[8] Available at: https://pypi.org/project/lda/
[9] Available at: http://scikit-learn.org/stable/index.html
[10] Available at: https://pypi.org/project/rake-nltk/

As for Ensemble MLP, the Top 5 best performance for combinations of embedding, optimizer and loss function when setting different thresholds are displayed in Table 7.

Table 7 Top 5 Best Performance for Combinations of Embedding, Optimizer and Loss Function with Different Thresholds

Combination	Threshold	Precision	Recall	F-measure
d2v-RMSprop-categorical_crossentropy	0.15	0.3830	0.6270	0.4760
d2v-adam-categorical_crossentropy	0.108	0.3665	0.6503	0.4688
d2v-RMSprop-binary_crossentropy	0.29	0.3736	0.6060	0.4622
d2v-adam-binary_crossentropy	0.248	0.3882	0.5643	0.4600
d2v-RMSprop-mean_squared_error	0.198	0.3683	0.6080	0.4587

Task 1B. Referring to the three different combination strategies using Model 1, Model 2 and Rule 1, Rule 2, Rule 3. We use the all the available data except testing data of CL-SciSumm 2019 and conduct the five fold cross validation over them. The average value of Precision (P), Recall (R) and F-measure (F) are shown in Table 8.

Table 8 Five Fold Cross Validation Result of Different Combinations of Model 1, Model 2 and Rule 1, Rule 2, Rule 3

Combination	Macro_P	Macro_R	Macro_F
$(Model\ 1, Rule\ 1, Rule\ 3)$	0.8248	0.8108	0.8128
$(Model\ 2\ , Rule\ 1, Rule\ 3)$	0.8493	0.8334	0.8374
$(Model\ 2\ , Rule\ 1, Rule\ 2, Rule\ 3)$	0.8633	0.8486	0.8517
Combination	Micro_P	Micro_R	Micro_F
$(Model\ 1, Rule\ 1, Rule\ 3)$	0.8223	0.7938	0.8076
$(Model\ 2\ , Rule\ 1, Rule\ 3)$	0.8498	0.8184	0.8337
$(Model\ 2\ , Rule\ 1, Rule\ 2, Rule\ 3)$	0.8604	0.8368	0.8483

4.3 Evaluation Results

Totally, we submit 30 different results. According to the released evaluations[7], there are different metrics to evaluate the system performance for task 1A and task 2. In Table 9, we display our highest value for each evaluation metric in each sub-task comparing with the best submissions of other teams. From Table 9, we can find that our team obtained two best performance among all the teams over the metric of task 1A-ROUGE-SU4 (F1) and task 2-Community R-SU4. Applied strategy for these run results are given below:

Task 1A. The best performance (Sentence Overlap) for task 1A applies the strategy of using new feature set. Compared with other strategies, the feature selection plays an important role. Although the results are much lower than the other teams, we think the training data set might be one of the reasons since SciSummNet is added into the training set. For metric ROUGE-SU4 in task 1A, there are 16 run results obtain the highest value. So most of the strategies proposed are effective for this evaluation.

Task 1B. The best performance for task 1B applies the strategy of combination $(Model\ 2\ , Rule\ 1, Rule\ 2, Rule\ 3)$, which also shows the best performance when

conducting over testing data (Table 8). Although task 1B is based on the results from task 1A, but the strategy of using bigger training data to train classifier and adding more rules truly work compared with the other strategies proposed in task 1B.

Task 2. The best performance for task 2 applies the strategy which is based on the identified facet obtained in task 1B. Such method to split identified cited text span into groups does show some logical evidences to infer the order of appearance for sentences.

Table 9 Highest Value for each Evaluation Metric in each Sub-task

Subtask	Metric	Our Best Submission	Best Submission of Other Teams
Task 1A	Sentence Overlap (F1)	0.086	0.126
	ROUGE-SU4 (F1)	***0.093***	0.09
Task 1B	F1	0.245	0.389
Task 2	Abstract R-2	0.296	0.514
	Abstract R-SU4	0.145	0.295
	Community R-2	0.204	0.209
	Community R-SU4	***0.117***	0.112
	Human R-2	0.237	0.278
	Human R-SU4	0.158	0.2

5 Conclusion

This paper introduces our submitted system IR&TM-NJUST at CL-SciSumm 2019. Compared with the previous work of task 1A, we constructed a new feature set based on correlation analysis, tried new negative sampling and added MLP-based models. For task 1B, a simple framework is proposed this year using classification model and prior rules. As to task 2, we make utilization of results from task 1B.

In the future work, more efforts can be done on these three tasks. For task 1, this year we have a larger data set thanks to SciSummNet. However, performance of classifiers that using this corpus is not good as expected. If possible, we should find more ways to expand available and valuable data set for this shared task. For task 2, except using the current pipeline, generative models can also be trained in the next work.

Acknowledgements

This work is supported by Major Projects of National Social Science Fund (No. 17ZDA291).

Reference

1. Jaoua, M. and A.B. Hamadou. *Automatic text summarization of scientific articles based on classification of extract's population*. in *International Conference on Intelligent Text Processing and Computational Linguistics*. 2003. Springer.

2. Abu-Jbara, A. and D. Radev. *Coherent citation-based summarization of scientific papers*. in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*. 2011. Association for Computational Linguistics.
3. Qazvinian, V. and D.R. Radev. *Scientific paper summarization using citation summary networks*. in *Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1*. 2008. Association for Computational Linguistics.
4. Elkiss, A., et al., *Blind men and elephants: What do citation summaries tell us about a research article?* Journal of the American Society for Information Science and Technology, 2008. **59**(1): p. 51-62.
5. Ma, S., et al. *NJUST @ CLSciSumm-17*. in *the Joint Workshop on Bibliometric-enhanced Information Retrieval and Natural Language Processing for Digital Libraries (BIRNDL 2017)*. 2017. Tokyo, Japan, CEUR.
6. Ma, S., et al. *NJUST@ CLSciSumm-18*. in *BIRNDL@ SIGIR*. 2018.
7. Chandrasekaran, M.K., et al. *Overview and Results: CL-SciSumm SharedTask 2019*. in *Proceedings of the 4th Joint Workshop on Bibliometric-enhanced Information Retrieval and Natural Language Processing for Digital Libraries (BIRNDL 2019) @ SIGIR 2019*. 2019. Paris, France.
8. Jaidka, K., et al. *The CL-SciSumm Shared Task 2017: Results and Key Insights*. in *BIRNDL@ SIGIR (2)*. 2017.
9. Jaidka, K., et al. *Overview of the CL-SciSumm 2016 shared task*. in *Proceedings of the Joint Workshop on Bibliometric-enhanced Information Retrieval and Natural Language Processing for Digital Libraries (BIRNDL)*. 2016.
10. Mayr, P., M.K. Chandrasekaran, and K. Jaidka. *Report on the 3rd Joint Workshop on Bibliometric-enhanced Information Retrieval and Natural Language Processing for Digital Libraries (BIRNDL 2018)*. in *ACM SIGIR Forum*. 2019. ACM.
11. Li, L., et al. *CIST@ CLSciSumm-18: Methods for Computational Linguistics Scientific Citation Linkage, Facet Classification and Summarization*. in *BIRNDL@ SIGIR*. 2018.
12. Davoodi, E., K. Madan, and J. Gu. *CLSciSumm Shared Task: On the Contribution of Similarity measure and Natural Language Processing Features for Citing Problem*. in *BIRNDL@ SIGIR*. 2018.
13. Baruah, G. and M. Kolla. *Klick Labs at CL-SciSumm 2018*. in *BIRNDL@ SIGIR*. 2018.
14. Wang, P., et al. *NUDT@ CLSciSumm-18*. in *BIRNDL@ SIGIR*. 2018.
15. Saeys, Y., et al., *Feature selection for splice site prediction: a new method using EDA-based feature ranking.* BMC bioinformatics, 2004. **5**(1): p. 64.
16. Lai, H., et al. *Greedy feature selection for ranking*. in *Proceedings of the 2011 15th International Conference on Computer Supported Cooperative Work in Design (CSCWD)*. 2011. IEEE.
17. Langley, P. and S. Sage, *Induction of selective Bayesian classifiers*, in *Uncertainty Proceedings 1994*. 1994, Elsevier. p. 399-406.
18. Guyon, I., et al., *Gene selection for cancer classification using support vector machines.* Machine learning, 2002. **46**(1-3): p. 389-422.
19. Li, L., et al. *CIST@ CLSciSumm-17: Multiple Features Based Citation Linkage,*

Classification and Summarization. in *BIRNDL@ SIGIR (2)*. 2017.
20. Ma, S., J. Xu, and C. Zhang, *Automatic identification of cited text spans: a multi-classifier approach over imbalanced dataset.* Scientometrics, 2018. **116**: p. 1303-1330.
21. Alonso, H.M., et al. *CL-SciSumm Shared Task-Team Magma*. in *BIRNDL@ SIGIR*. 2018.
22. Prasad, A. *WING-NUS at CL-SciSumm 2017: Learning from Syntactic and Semantic Similarity for Citation Contextualization*. in *BIRNDL@ SIGIR (2)*. 2017.
23. Agrawal, K. and A. Mittal. *IIIT-H@ CLScisumm-18*. in *BIRNDL@ SIGIR*. 2018.
24. De Moraes, L.F., et al. *University of Houston@ CL-SciSumm 2018*. in *BIRNDL@ SIGIR*. 2018.
25. Abura'ed, A., et al. *LaSTUS/TALN+ INCO@ CL-SciSumm 2018-Using Regression and Convolutions for Cross-document Semantic Linking and Summarization of Scholarly Literature*. in *Proceedings of the 3nd Joint Workshop on Bibliometric-enhanced Information Retrieval and Natural Language Processing for Digital Libraries (BIRNDL2018). Ann Arbor, Michigan (July 2018)*. 2018.
26. Debnath, D., A. Achom, and P. Pakray. *NLP-NITMZ@ CLScisumm-18*. in *BIRNDL@ SIGIR*. 2018.
27. Mihalcea, R. and P. Tarau. *Textrank: Bringing order into text*. in *Proceedings of the 2004 conference on empirical methods in natural language processing*. 2004.
28. Pearson, K., *VII. Note on regression and inheritance in the case of two parents.* proceedings of the royal society of London, 1895. **58**(347-352): p. 240-242.
29. Kingma, D.P. and J. Ba, *Adam: A method for stochastic optimization.* arXiv preprint arXiv:1412.6980, 2014.
30. Yasunaga, M., et al., *ScisummNet: A Large Annotated Corpus and Content-Impact Models for Scientific Paper Summarization with Citation Networks.* 2019.