Argument Component Classification by Relation Identification by Neural Network and TextRank

Mamoru Deguchi
University of Tokyo
deguchi-mamoru@graco.c.u-tokyo.ac.jp

Kazunori Yamaguchi
University of Tokyo
yamaguch@graco.c.u-tokyo.ac.jp

Abstract

In recent years, argumentation mining, which automatically extracts the structure of argumentation from unstructured documents such as essays and debates, is gaining attention. For argumentation mining applications, argument-component classification is an important subtask.

The existing methods can be classified into supervised methods and unsupervised methods. Many existing supervised methods use a classifier to identify the roles of argument components, such as " claim " or " premise " , but many of them use information of a single sentence without relying on the whole document. On the other hand, existing unsupervised document classification has the advantage of being able to use the whole document, but accuracy of these methods is not so high.

In this paper, we propose a method for argument-component classification that combines relation identification by neural networks and TextRank to integrate relation informations (i.e. the strength of the relation). This method can use argumentation-specific knowledge by employing supervised learning on a corpus while maintaining the advantage of using the whole document.

Experiments on two corpora, one consisting of student essays and the other of Wikipedia articles, show the effectiveness of this method.

1 Introduction

In recent years, argumentation mining, which automatically extracts the structure of argumentation from unstructured documents such as essays and debates, is gaining attention.

Argumentation mining consists of the following four subtasks (Potash et al., 2017).

1. Extracting the argument component (AC for short) from a given document (component idenfication).
2. Assigning a label such as claim or premise to each AC (component classification).
3. Determining whether each pair of ACs is related or not (relation identification).
4. Assigning a label such as attack or support to the related pairs of ACs (relation classification).

We focus on component classification. Generally, an AC is classified as a claim or a premise. A claim is the conclusion of an argument, and a premise is an assumption or reason to induce the conclusion.

In this paper, we propose a method that can be applied to new domains of argumentation with little cost.

Previous methods of component classification can be classified as supervised document classification and unsupervised document classification using topic models or TextRank, a ranking algorithm. Many existing methods of supervised document classication perform classication using a single sentence without relying on the whole document. Existing unsupervised document classification has the advantage of being able to use the whole document, but it can not use argumentation specific knowledge, such as the fact that "therefore" relates a conclusion with its reason.

In this paper, we propose a framework for component classification using the argumentation-specific knowledge by employing supervised learning on a corpus while maintaining the advantage of using the whole document.

The research on component classification has been done on various domains of argumentation. (Levy et al., 2014) proposed a method to extract from a Wikipedia artitle a sentence including Context Dependent Claim (CDC) that directly supports the topic of the article by combining contextfree and context-dependent features such as the cosine similarity between the topic of the article and the sentence. (Lippi and Torroni, 2015) proposed a method to extract a CDC from a sentence using a support-vector machine (SVM) with Tree Kernels on the phrase structure of the sentence. As for student essays, (Stab and Gurevych, 2014) proposed a method to classify a sentence as a major claim, claim, or premise using an SVM on the basis of features such as the position of the AC in a paragraph and a clue expression. However, these methods have a disadvantage in that new features must be developed in order to apply said methods to a new domain of argumentation.

(Daxenberger et al., 2017) proposed a method to extract claims in various domains of argumentation using a recurrent neural network (RNN) and convolutional neural network (CNN). These are methods for classifying a sentence, so they cannot use information outside of the sentence. This is a significant disadvantage for these methods in component classification because the role of an AC is determined relatively to those of other ACs in the document. For example, the probability that an AC is a claim is higher if the AC is supported by some premises. Thus, using the relation of an AC to other ACs is important when classifying the AC.

Some researchers have tried to improve the performance of component classification by employing the relation information between ACs. In (Stab and Gurevych, 2017), the results of component classification and relation identification are combined to improve both of their performances using integer programming. In this method, the cost to apply a new domain of argumentation is high because hand-crafted features are highly dependent on the domain of argumentation such as the position of AC in the argumentation or a clue expression.

In (Potash et al., 2017), component classification and relation identification between ACs are performed simultaneously using a PointerNet neural network. This improves the classification performance. In this research, the dependency relation is limited to within a paragraph, and the whole document cannot be used.

There is also research on component classification using unsupervised learning. (Ferrara et al., 2017) proposed a method to extract a sentence including AC, extract a major claim (the standpoint of the author for the topic of an essay), and classify ACs using a topic model. In (Petasis and Karkaletsis, 2016), sentences including a major claim and a claim are extracted by ranking the sentences using the TextRank (Mihalcea and Tarau, 2004) on the basis of the similarity of sentences. These studies use relations among ACs in a document and are not dependent on the domain of argumentation. However, these methods are not highly accurate.

In this paper, we propose a neural network to evaluate the probability of there being a relation between ACs and to rank ACs using TextRank on the basis of probability.

Our method uses argumentation-specific knowledge for relation identification between ACs, and the results are used for component classification. The argumentation-specific knowledge is extracted by a neural network from a small corpus. Thus, this method can be applied to various domains of argumentation with little cost. We applied the proposed method to two domains of argumentation and had positive results.

2 Previous Methods

2.1 Component Classification using TextRank

In this section, we explain what TextRank is and discuss previous methods of component classification using TextRank.

TextRank is a PageRank-based ranking algorithm applied to natural language processing. It has been used for keyword extraction and extraneous document summarization. In TextRank, a document is represented by a weighted directed graph with a fragment of a text such as a sentence, phrase, or word as a node; a metric between two nodes are used as a weight on the edge between the nodes. From this directed graph, a recurrent equation is generated and its solution is used to determine the rank of the nodes.

For example, suppose that a weighted direct graph $G = (S, E)$ is obtained from a document $D = \{S_1, ..., S_n\}$. Here, $E \subseteq S \times S$ and E_{ij} $(1 \leq i, j \leq n)$ are a directed edge from sentence S_i to sentence S_j. For a directed edge E_{ij}, a metric $w(S_i, S_j)$ from a sentence S_i to S_i is used as the weight of the edge. Then, $WS(S_i)$ determined by Eq. 2 is used as the score of the sentence S_i.

$$W(S_i, S_j) = \frac{w(S_i, S_j)}{\sum_{S_k \in Out(S_i)} w(S_i, S_k)} \tag{1}$$

$$WS(S_i) = (1-d)+d* \sum_{S_j \in In(S_i)} W(S_j,S_i) \cdot WS(S_j) \quad (2)$$

$In(S_i)$ is the set of sentences that have an outgoing edge to the sentence S_i, and $Out(S_j)$ is the set of sentences that have an incoming edge from S_j. d is a hyperparameter (random surfer rate) taking a value between 0 and 1. In the previous work (Petasis and Karkaletsis, 2016; Ferrara et al., 2017), the term frequency inverse document frequency (TFIDF) cosine similarity between sentences S_i and S_i was used as $w(S_i, S_j)$. In (Petasis and Karkaletsis, 2016), the score was determined for each sentence, and the method was evaluated correct if the top one or two sentences according to the score includes the target major claim or claim.

2.2 Relation Identification between Argument Components

In this section, we give an overview of research on relation identification between ACs. (Stab and Gurevych, 2017) performed a binary classification of whether or not there is a relation between two ACs using SVM on the basis of features extracted from ACs in the domain of student essays. (Nguyen and Litman, 2016) performed a classification of the relation between ACs using features obtained from the information around the AC as well as the features obtained from the AC itself. (Rinott et al., 2015) classified evidence (claim dependent evidence: CDE) into Study, Expert, and Anecdotal in accordance with their properties in Wikipedia articles of a random topic, and then extracted CDE of each Claim using a combination of logistic regression (LR) classifiers. In these studies, features used for classification have to be prepared by hand, so these methods have the disadvantage of a high cost to develop the features.

(Cocarascu and Toni, 2017) classified the relation between ACs with neural networks using a corpus they developed.

In this paper, we propose a method that can be applied to various domains of argumentation with little cost. We think that preparing a small corpus with labels is acceptable for better accuracy, but developing new features is too costly because for developing new features, skills on feature enginering as well as knowledge on the domaim of argument are required. For these reasons, we employ a novel neural network to determine the probability that an AC is related to other ACs as in the approach of (Cocarascu and Toni, 2017), use probability as the weight for TextRank, and use the score of the AC as the likelihood that the AC is major claim or claim.

3 Proposed Method

In this section, we explain our approach. Section 3.1 explains the neural network we use to identify the relations between ACs. Section 3.2 explains a method for extracting claims by applying the TextRank algorithm to the identified relation.

3.1 Relation Identification

We use a neural network for identifying the relations between ACs. The neural network is used to convert an AC into a single sentence vector and to output the probability that there exists a relation between a pair of ACs using the vectors of the ACs. The neural network consists of

1. a neural network to convert an AC into a single sentence vector, and
2. a neural network to assess the relatedness of the vectors of two ACs.

We tested long short-term memory (LSTM) and a CNN for Step 1. In Step 2, we tested the following two methods to combine the vectors of two documents obtained in Step 1. In the first method, we concatenated the two vectors and fed them to the fully connected layer. In the second method, we fed the two vectors as a sequence to LSTM, and the hidden units of LSTM were concatenated and sent to the fully connected layer.

We evaluated the neural networks on the basis of their performance when combined with TextRank. Figure 1 shows the neural network that obtained the best performance.

The input to the neural network is a pair consisting of $AC_i = (w_1, w_2, ...w_k)$ and $AC_j = (w'_1, w'_2, ...w'_{k'})$, where w_l is a word in an AC and k and k' is the length of the AC_i and AC_j. AC_i and AC_j are converted into word vectors $V_i = (v_1, v_2, ...v_k)$ and $V_j = (v'_1, v'_2, ...v'_{k'})$ in the embedding layer. v_l is a word vector for a word w_l. They are transformed by LSTM to make sentence vectors V_{AC_i} and V_{AC_j}. V_{AC_i} and V_{AC_j} are concatenated and sent to the next LSTM layers. Then, hidden state of each timesteps of LSTM layer are concatenated and sent to the next dense layers. Finally, the softmax function produces the estimated probability of the relation of the pair.

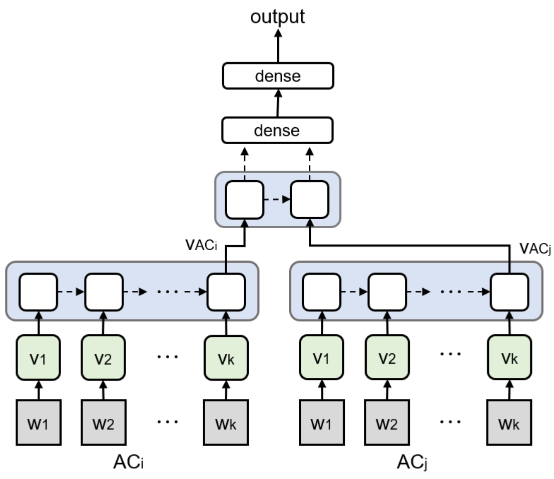

Figure 1: Proposed neural network for relation identification: LSTM and dense network

If there is a relation between AC_i and AC_j, we denote the pair as $link$, and otherwise as $nolink$.

3.2 TextRank for Argument Mining

The relation identifier in Section 3.1 yields the probability $P(link|S_i, S_j)$ from AC_i to AC_j. We use this estimated probability $P(link|S_i, S_j)$ as $w(S_i, S_j)$ in Eq. 1. Because there is no ground to determine $w(S_i, S_i)$, we empirically set $w(S_i, S_i)$ to 1.

4 Experiment

In this section, we explain the corpus we use, experimental setting, and results.

4.1 Data

We used two copora in this experiment, one consisting of student essays and the other of Wikipedia articles.

4.1.1 Student Essay

(Stab and Gurevych, 2017) distributed annotated student essays in English extracted from the online forum essayforum.com at https://www.informatik.tu-darmstadt.de/ (Argument Annotated Essays (version 2)). The annotation consists of classification labels of the ACs and the relations among the ACs. We call this corpus "Student Essay." The basic figures of Student Essay are shown in Table 1.

As for the classification, ACs are classified as major claims, claims, and premises. A major claim shows the standpoint of the author on the topic of the essay. A claim supports or attacks the major claim. A premise is an assumption or a reason in an argument and supports or attacks a claim or another premise.

Regarding relations between ACs, the relation between claims is generally marked as for or against and that between premises as support or attack. In this paper, however, we do not use these types of relation (i.e. for, against, support, or attack). Rather, if there exists a relation of any type between a pair of ACs, we consider the pair as a positive example. Other pairs of ACs could serve as negative examples. However, the number of such negative examples is much larger than that of positive examples. In addition, most of the negative example AC pairs are irrelevant to learning. Thus, we used only the reverse pairs (i.e. major claim and claim, major claim and premise, and claim and premise) as negative examples.

4.1.2 Wikipedia Article

(Aharoni et al., 2014) distributed annotated Wikipedia articles[1] with the topic labels, claim (CDC), and context dependent evidence (CDE). A topic is a short statement of the subject of an article. CDC is a statement supporting or attacking the topic that is directly related to a main claim of the article. CDE is a text fragment directly supporting some CDC under the topic of the article. We call this corpus "Wikipedia Article." The basic figures of Wikipedia Article are shown in Table 2.

CDE can be classified as Study, Expert, and Anecdotal according to the type of evidence. Study is CDE backed by quantitative analysis. Expert is CDE backed by an expert (person or organization). Anecdotal is CDE backed by an event or example. In this experiment, if CDC and CDE in an article were related, we used the pair as a positive example. Otherwise, they were used as a negative example. We did not use the different types of CDE.

Table 1: Student Essay

essay	Type of AC			Relation	
	MajorClaim	Claim	Premise	link	nolink
402	751	1506	3832	6673	91798

4.2 Proposed Method

In this section, we explain the experimental details of the proposed method. For

[1] http://www.research.ibm.com/haifa/dept/vst/debating_data.shtml

Table 2: Wikipedia Article

essay	Type of AC		Relation	
	CDC	CDE	link	nolink
102	350	795	1291	31634

word vectors, we used pretrained word vectors with 300 dimensions available at https://code.google.com/archive/p/word2vec/.

The neural network was implemented in Keras[2].

For Student Essay, we used the following settings. LSTM had 64 units. The fully connected layer had 64 units with a dropout rate of 0.5. A sigmoid function was used in the output layer. Binary cross-entropy was used as the loss function. The batch size was 128. Early stopping was employed using validation loss. The longest AC consisted of 67 words with 7238 words in vocabulary. We employed five-fold cross validation for testing. Ten percent of the training data was used as validation data.

For Wikipedia Article, we used the following settings. LSTM had 32 units. The fully connected layer had 64 units with a dropout rate of 0.3. A sigmoid function was used in the output layer. Binary cross-entropy was used as the loss function. The batch size was 128. Early stopping was employed using validation loss. The longest AC was 254 words with 6412 words in vocabulary. We employed ten-fold cross validation for testing because the data size of Wikipedia Article is smaller than that of Student Essay. Ten percent of the training data was used as validation data.

The hyperparameter d of TextRank was set 0.85 throughtout the experiments.

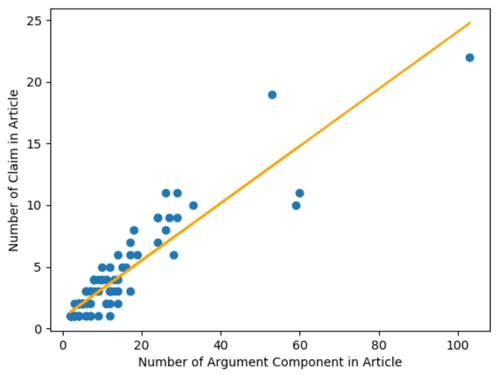

Figure 2: The Number of ACs vs. That of Claims in Wikipedia Article

[2]https://keras.io/

4.3 Previous Work: TextRank-TFIDF and TextRank-W2V

Here, we explain the similarities used in TextRank for comparison. For the similarity of TFIDF, we used the TFIDF vectors as used in (Petasis and Karkaletsis, 2016). We call this TextRank with the TFIDF cosine similarity "TextRank-TFIDF".

For the similarity of word2vec, we used the vector obtained by averaging the vectors of words in a sentence. For the word vectors, we used pre-trained Word2Vec with 300 dimensions at https://code.google.com/archive/p/word2vec/. We call TextRank with the word2vec consine similarity "TextRank-W2V".

4.4 Previous Work: Supervised Component Classification

Here, we explain the detail of supervised component classification for comparison. Because Student Essay has major claims and claims, we constructed two classifiers: the one which detects only major claims and another which detects all claims (including the major claims). For the classifier, we used a neural network classifier. As for the neural networks, we tested LSTM, bidirectional LSTM (biLSTM), and CNN. The input to the network was a sequence of AC word vectors. The output was whether the input AC is major claim (claim) or not. We used the following settings. LSTM and biLSTM had 64 units. The fully connected layer had 64 units with dropout rate 0.5. A sigmoid function was used in the output layer. The binary crossentropy was used as the loss function. The CNN had the kernel sizes 3, 4, and 5. The number of filters was 64. The max pooling had poolsize 2. The fully connected layer had 64 units with dropout rate 0.5. A sigmoid function was used in the output layer. Binary crossentropy was used as the loss function.

To discriminate major claims, claims, and premises, we used three-way classification. For three-way classification, we employed the softmax function. For the loss, we used categorical cross entropy.

We employed five-fold cross validation for testing. Ten percent of the training data was used as validation data for early stopping.

For word vectors, we used the pre-trained Word2Vec with 300 dimensions available at https://code.google.com/archive/p/word2vec/.

Table 3: Results of Claim Detection using TextRank in Student Essay

		TextRank-TFIDF		TextRank-W2V		**Our Model**	
evaluation metrics	total	correct essay	accuracy	correct essay	accuracy	correct essay	accuracy
MajorClaim@1	402	116	0.289	79	0.197	218	**0.542**
MajorClaim@2	402	173	0.430	128	0.318	282	**0.701**
MajorClaim@3	402	215	0.535	176	0.438	326	**0.811**
Claim@1	402	252	0.627	198	0.493	319	**0.794**
Claim@2	402	330	0.821	291	0.724	372	**0.925**
Claim@3	402	361	0.898	337	0.838	392	**0.975**

Table 4: Results of Claim detection using TextRank in Wikipedia Article

		TextRank-TFIDF		TextRank-W2V		**Our Model**	
evaluation metrics	total	correct essay	accuracy	correct essay	accuracy	correct essay	accuracy
Claim@1	104	13	0.125	2	0.019	101	**0.971**
Claim@2	104	44	0.423	30	0.288	103	**0.990**

Table 5: Averaged Rank of Major Claim, Claim, and Premise in Student Essay

Component Type	TextRank-TFIDF	TextRank-W2V	Our Model
MajorClaim	6.397	7.071	**3.883**
Claim	7.609	8.050	**6.664**
Premise	9.399	9.093	**10.263**

Table 6: Average Rank of Claim and Premise in Wikipedia Article

Component Type	TextRank-TFIDF	TextRank-W2V	Our Model
Claim	17.162	17.780	**4.747**
Premise	14.368	14.089	**19.959**

4.5 Evaluation Method

For the comparison between our method and TextRank-TFIDF/TextRank-W2V, we used Claim@k and MajorClaim@k as evaluation metrics. In MajorClaim@k, the result is considered correct if the top k according to the ranking includes the target major claim. In Claim@k, the result is considered correct if the top k according to the ranking includes the target claim and major claim. In Student Essay, MajorClaim@k and Claim@k were evalueted for $k = 1, 2, 3$. Wikipedia Article does not include major claim, so the evaluation was done only for Claim@k. The number of ACs varies significantly, and the minimum is 2, so we report Claim@k for $k = 2$ to evaluate all the ACs. The number of articles that had two ACs was 24 out of 102. This means that these 24 articles are considered correct regardless of the output of the classifier when evaluating at Claim@2. So we should be careful that there is possibility of overestimation.

For component classification, we employed the precision, recall, and F-score as evaluation metrics that are often used in text classification. Our method obtains a rank of the ACs. For comparison, we set a threshold; if the rank was higher than the threshold, we considered the AC to be a major claim or claim. We used 1-3 as the threshold for Student Essay.

The number of ACs varies more for Wikipedia Article, ranging from a minimum of 2 to maximum of 103, with an average of 11.12. Thus, if we were to employ a small, fixed threshold, the recall would get smaller for an article with many ACs. In order to resolve this problem, we employed a linear regression to predict the number of claims from the number of ACs and used the prediction as a threshold. We call this "@adaptive." For the regression on Wikipedia Article, the regression coefficient was 0.232, and the intercept was 0.873 with R-squared as 0.846. The fitted line is drawn

Table 7: Precision and Recall detecting Claim in Student Essay

Method	Precision	Recall	F-Score
Claim@1	**0.794**	0.146	0.247
Claim@2	0.748	0.276	0.403
Claim@3	0.715	0.395	0.509
Claim@6	0.602	0.661	0.630
Claim@7	0.571	**0.731**	**0.641**
LSTM	0.60	0.62	0.61
BiLSTM	0.57	0.61	0.59
CNN	0.58	0.58	0.58

Table 8: Precision and Recall detecting Major Claim in Student Essay

Method	Precision	Recall	F-Score
MajorClaim@1	**0.542**	0.298	0.384
MajorClaim@2	0.437	0.472	0.454
MajorClaim@3	0.371	**0.602**	**0.458**
LSTM	0.44	0.39	0.41
BiLSTM	0.49	0.35	0.41
CNN	0.49	0.31	0.38

Table 9: Precision and Recall detecting Claim in Wikipedia Corpus

Method	Precision	Recall	F-Score
Claim@adaptive	0.832	0.875	0.853
Claim@1	**0.971**	0.554	0.706
Claim@2	0.788	0.734	0.760
LSTM	0.89	**0.97**	**0.93**
BiLSTM	0.92	0.86	0.90
CNN	0.94	0.91	**0.93**

in Fig. 2.

For Student Essay, we evaluated a case wherein only major claims is considered and one wherein both major claims and claims are considered.

4.6 Experimental Result and Discussion

Tables 3 and 4 show the results using our method and TextRank-TFIDF/TextRank-W2V to Student Essay and Wikipedia Article.

For Student Essay, our proposed method outperformed the previous TextRank-TFIDF/TextRank-W2V. In particular, our method achieved 0.542, which is significantly better than the 0.289 of TextRank-TFIDF, for major claims.

Simply employing word vectors alone did not improve the performance; MajorClaim@1 was 0.197 for TextRank-W2V while it was 0.289 for TextRank-TFIDF.

Table 5 shows the averaged rank of major claims, claims and premises. For TextRank-TFIDF/TextRank-W2V, the difference in the averaged ranks for major claims, claims, premises is small, and they are not well separated. In our proposed method, the averaged ranks of major claims, claims, and premises are 3.883, 6.664, and 10.263, respectively, and they are well articulated. For Wikipedia Article, our method correctly assigns a higher rank to claims while TextRank-TFIDF/TextRank-W2V incorrectly assign a higher rank to premises.

Tables 7, 8, and 9 show the result of comparison of our method to the neural network classifiers. Our method ranks ACs into specified types of AC: major claim, claim, or premise for Student Essay. Because the neural network classifiers are classifiers, in order to make a comparison, we set a threshold on the rank to make classification. For Wikipedia Article, because the number of ACs varies, we use an adaptive threshold explained in Section 4.5.

For Student Essay, our method was the best in F-Score for major claim with 3 as the threshold, and also for claim with 7 as seen in Tables 7 and 8. This shows the effectiveness of our method for classifying AC into major claim and claim.

Table 10: F-Score of three-way Classification of MajorClaim, Claim, and Premise of Student Essay

Method	MajorClaim	Claim	Premise
LSTM	0.37	0.34	0.75
BiLSTM	0.27	0.22	0.76
CNN	0.33	0.30	0.75

Table 11: Confusion Matrix of three-way Classification of MajorClaim, Claim, and Premise using LSTM of Student Essay for 20% test set of Table 1

	MajorClaim	Claim	Premise
MajorClaim	44	44	71
Claim	18	96	201
Premise	19	113	612

For Student Essay, our method is better than the neural network classifier. Table 10 shows the F-score for three-way classifier. In this table, LSTM and biLSTM are slightly better than CNN, but the difference is small. It is notable that the score is high for premise, but it is low for major claim and claim. Table 11 shows the confusion matrix for LSTM. The table shows this more clearly. Our method is effective for discriminating major claim and claim because major claim and claim are separated by ranking,

For Wikipedia Article, the neural network classifier marked better F-Scores. This can be understood that the argumentation structure of Wikipedia Article is more controlled and can be extracted just by the neural network classifier. We show the confusion matrix for LSTM of Wikipedia Article at Table 12.

However our method is better than the neural network classifier in the precision of @1 in Table 9. If one want to find out not all the claim but main claim, our method can serve better.

In summary, our method outperformed the previous methods for Student Essay. For Wikipedia Article, our method was slightly worse than the neural network classifiers.

5 Conclusion and Future Work

In this paper, we proposed a method to classify claim (major claim) by the combination of the neural network to determine the relation between ACs and TextRank to integrate the relation information to rank claim (major claim) higher. The

Table 12: Confusion Matrix of Classification of Claim, and Premise using LSTM of Wikipedia Article for 20% test set of Table 2

	Claim	Premise
Claim	70	2
Premise	9	151

experiments on Student Essay and Wikipedia Article show that the proposed method performed better in major claim and claim classification compared with TextRank with unsupervised similarity measure. This shows the effectiveness of utilizing the relation between ACs. Compared with the neural network classifier, the proposed method performed better for Student Essay, and not better in F-Score but better in precision for Wikipedia Article. Thus, if we need more precision such as the case that we want to find out only claim, the proposed method has an advantage. In addition, the proposed method performed well for a rather complex argument structure such as major claim, claim, and premise utilizing the ranking produced by the method. In summary, the proposed method performed well for multiple corpora with different argument structures and varying number of ACs.

We are going to use word vectors using the contextual information to improve the relation identification and also test other ranking algorithms such as RankNet (Burges et al., 2005) and ListNet (Cao et al., 2007).

References

Ehud Aharoni, Anatoly Polnarov, Tamar Lavee, Daniel Hershcovich, Ran Levy, Ruty Rinott, Dan Gutfreund, and Noam Slonim. 2014. A benchmark dataset for automatic detection of claims and evidence in the context of controversial topics. In *Proceedings of the First Workshop on Argumentation Mining*, pages 64–68, Baltimore, Maryland. Association for Computational Linguistics.

Chris Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. 2005. Learning to rank using gradient descent. In *Proceedings of the 22Nd International Conference on Machine Learning*, ICML '05, pages 89–96, New York, NY, USA. ACM.

Zhe Cao, Tao Qin, Tie-Yan Liu, Ming-Feng Tsai, and Hang Li. 2007. Learning to rank: From pairwise approach to listwise approach. In *Proceedings of the 24th International Conference on Machine Learning*, ICML '07, pages 129–136, New York, NY, USA. ACM.

Oana Cocarascu and Francesca Toni. 2017. Identifying attack and support argumentative relations using deep learning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1374–1379, Copenhagen, Denmark. Association for Computational Linguistics.

Johannes Daxenberger, Steffen Eger, Ivan Habernal, Christian Stab, and Iryna Gurevych. 2017. What is the essence of a claim? cross-domain claim identification. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2055–2066, Copenhagen, Denmark. Association for Computational Linguistics.

Alfio Ferrara, Stefano Montanelli, and Georgios Petasis. 2017. Unsupervised detection of argumentative units though topic modeling techniques. In *Proceedings of the 4th Workshop on Argument Mining*, pages 97–107, Copenhagen, Denmark. Association for Computational Linguistics.

Ran Levy, Yonatan Bilu, Daniel Hershcovich, Ehud Aharoni, and Noam Slonim. 2014. Context dependent claim detection. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 1489–1500, Dublin, Ireland. Dublin City University and Association for Computational Linguistics.

Marco Lippi and Paolo Torroni. 2015. Context-independent claim detection for argument mining. In *Proceedings of the 24th International Conference on Artificial Intelligence*, IJCAI'15, pages 185–191. AAAI Press.

Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into text. In *Proceedings of EMNLP 2004*, pages 404–411, Barcelona, Spain. Association for Computational Linguistics.

Huy Nguyen and Diane Litman. 2016. Context-aware argumentative relation mining. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1127–1137, Berlin, Germany. Association for Computational Linguistics.

Georgios Petasis and Vangelis Karkaletsis. 2016. Identifying argument components through textrank. In *Proceedings of the Third Workshop on Argument Mining (ArgMining2016)*, pages 94–102, Berlin, Germany. Association for Computational Linguistics.

Peter Potash, Alexey Romanov, and Anna Rumshisky. 2017. Here's my point: Joint pointer architecture for argument mining. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1364–1373, Copenhagen, Denmark. Association for Computational Linguistics.

Ruty Rinott, Lena Dankin, Carlos Alzate Perez, Mitesh M. Khapra, Ehud Aharoni, and Noam Slonim. 2015. Show me your evidence - an automatic method for context dependent evidence detection. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 440–450, Lisbon, Portugal. Association for Computational Linguistics.

Christian Stab and Iryna Gurevych. 2014. Identifying argumentative discourse structures in persuasive essays. In *EMNLP*, pages 46–56.

Christian Stab and Iryna Gurevych. 2017. Parsing argumentation structures in persuasive essays. *Computational Linguistics*, 43(3):619–659.