Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 116 (2017) 449–459

Procedia Computer Science

www.elsevier.com/locate/procedia

2nd International Conference on Computer Science and Computational Intelligence 2017, ICCSCI 2017, 13-14 October 2017, Bali, Indonesia

It Takes Two To Tango: Modification of Siamese Long Short Term Memory Network with Attention Mechanism in Recognizing Argumentative Relations in Persuasive Essay

Aryo Pradipta Gema[a], Suhendro Winton[a], Theodorus David[a], Derwin Suhartono*[a], Muhsin Shodiq[a], Wikaria Gazali[b]

[a]*Computer Science Department, School of Computer Science, Bina Nusantara University,*
Jl. K. H. Syahdan No. 9 Kemanggisan, Jakarta 11480, Indonesia
[b]*Mathematics Department, School of Computer Science, Bina Nusantara University,*
Jl. K. H. Syahdan No. 9 Kemanggisan, Jakarta 11480, Indonesia

Abstract

We propose a novel approach in a dataset of argumentation relations. This task is intended to analyze the presence of a support relation between two sentences. To be able to identify relations between two sentences or arguments, one is obliged to understand the nuance brought by both sentences. Our models are modification of siamese network architectures, in which we replace the feature extractor into Long Short Term Memory and implement cosine distance as the energy function. Our models take a pair of sentences as their input and try to identify whether there is a support relation between those two sentences or not.The primary motivation of this research is to prove that a high degree of similarity between two sentences correlates to sentences supporting each other. This work will focus more on the modification of siamese network and the implementation of attention mechanism. Due to the difference in dataset setting, we cannot arbitrarily compare our results with the prior research results. Therefore, this work will not highlight the comparison between deep learning and traditional machine learning algorithm per se, but it will be more of an exploratory research. Our models are able to outperform the baseline score of accuracy with a margin of 17.33% (67.33%). By surpassing the baseline performance,we believe that our work can be a stepping stone for deep learning implementation in argumentation mining field.

© 2017 The Authors. Published by Elsevier B.V.
Peer-review under responsibility of the scientific committee of the 2nd International Conference on Computer Science and Computational Intelligence 2017.

Keywords: Siamese Network; Long Short Term Memory, Attention Mechanism, GloVe, Cosine Distance

* Corresponding author. Tel.: +62-215345830
E-mail address: dsuhartono@binus.edu

1877-0509 © 2017 The Authors. Published by Elsevier B.V.
Peer-review under responsibility of the scientific committee of the 2nd International Conference on Computer Science and Computational Intelligence 2017.
10.1016/j.procs.2017.10.036

1. Introduction

Comprehensively understand an argument or even a part of it is not an easy task. It requires the reader to understand each word that build the sentence or part of an argument itself. Not only to understand each word meaning, but also understand the relation between one word and another. Argumentation understanding does not stop only on the word level, it is more of a hierarchical problem. One should also understand the relation between two sentences. This is the fundamental idea why argumentation can actually represent the unreasonable intellectual capacity of human. An attempt to copy that ability is the foundation of argumentation mining researches.

As expected, many showed interest on argumentation mining, even though it is still a relatively young research field. Many researches forged novel approaches and achieved the state-of-the-art results. Several researches in identifying argument components relied heavily on handcrafted features[1 2]. They managed to observed several machine learning algorithms in conjunction with handcrafted features, such as contextual features and discourse marker. The notable work to identify argumentative discourse structures also relied on handcrafted features[3]. In that work, they also managed to introduce the tree of argument structure in which revolutionizes our understanding of argument structure. Taking the benefit of the tree, they introduced parse features by calculating the depth of the tree and the number of subclasses.

Deep learning algorithms have revolutionized machine learning paradigm in the last decade. Instead of spending hours to engineer features that are most suitable for a particular task, one can conveniently inject raw data to a deep learning architecture and get a comparable, if not better, results. Deep learning architectures are made ranging from Computer Vision to Natural Language Processing (NLP) domain to extract their high level features that might not be seen by human experts. A research in slot filling task proved that deep learning can perceive 42% more unseen features in comparison to handcrafted features[4].

Deep learning breakthroughs did help researches in Natural Language Processing (NLP) tasks. The invention of word vector representation[5 6] reformed the way researchers quantify words. The development of Recurrent Neural Network (RNN)[7] and its modifications[8 9] and Convolutional Neural Network (CNN) in sentence classification[10] are the state-of-the-art algorithms in Natural Language Processing (NLP). Several recent argumentation mining researches are attempted to follow the deep learning trend as well. In 2016, an attempt[11] used Bidirectional Long Short Term Memory architecture to compare the level of convincingness of two arguments. A more recent work[12], implemented CNN to recognize insufficiently supported arguments. Another work focused on the implementation of word vector representation and Long Short Term Memory unit to identify argument components[13].

This paper attempted to focus more on the exploration of deep learning in a task which dataset is released and havent tried yet. We adopt the latest publicly available dataset of argumentation structure[14] which consists of 402 essays and do several experiments on the argument relations task. As the highlight, we implemented Long Short Term Memory (LSTM) siamese network with cosine proximity as the energy function to identify argumentative relations between pairs of sentences. We argue that when a sentence supports another sentence, there must be a certain degree of similarity between both sentences and we believe that cosine proximity can represent that similarity metric better than other reasonable alternatives. In this paper, we presented results of our model that can outperform the baseline of this research. As a disclaimer, this will not be comparison experiments towards the prior research results[1], considering that we used a different dataset. Instead, this is more of an analysis of the deep learning algorithm in argumentation mining field.

2. Related Works

Identifying argumentative relations is a part of argumentation mining tasks. In order for one to identify an argumentative relation between two sentences, one needs to comprehensively understand the semantic substance of both sentences. That’s why this task can be classified as an argument analysis subdomain. Extensive researches in argument analysis subdomain are exponentially increasing nowadays. For example, automatic argumentation summarizer in a political debate corpus[15] which requires an understanding of the entire debate implicit meaning. An attempt to automatic convincingness level prediction and ranking by using a pairwise learning process Support Vector Machine (SVM) and Bidirectional Long Short Term Memory (BiLSTM)[11]. This task requires the machine to understand the nuance of each argument before it can decide which argument is more convincing.

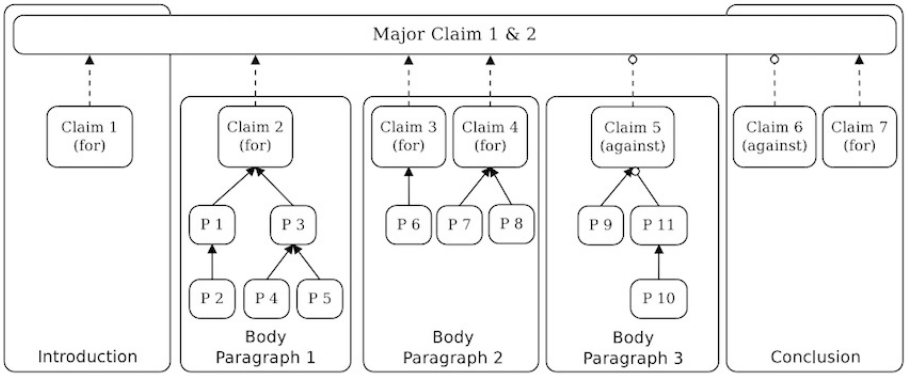

Fig. 1. Argumentation structure of the example essay. Every edge symbolize argumentative relations. Edges that has an arrowhead or a circlehead denote argumentative support or attack relations respectively. Dashed lines denote relations that are encoded in the stance attributes of claims, which is not used in this task. P stands for premises [14]

Besides argument analysis tasks, identifying argumentative relations task is closely related to discourse analysis tasks. For instance, a task to identify implicit discourse relations [16]. They collected corpus and automatically labelled the data according to the discourse marker in the second sentence (i.e. *'because'* or *'but'*). The most related work to ours is also to identify argument relations [1]. They managed to create an annotated corpus which consists of 90 essays, and did some experiments using traditional machine learning classifiers such as, SVM, Naive Bayes, C4.5 Decision Tree and Random Forest. With a plethora of features combination, SVM yielded the best results for this task. The first attempt to implement siamese network is for a signature verification task [17]. In that task, they create two identical network that will extract the features from two images in parallel that later the difference between those images will be calculated and used as the metrics of learning. Siamese network has also been used for face verification task [18]. They introduced a novel loss function for siamese network, which is contrastive loss. Siamese networks have also been proposed for various metric learning tasks [19 20]. Unfortunately, to the best of our knowledge, siamese networks with recurrent unit remain largely unexplored. The only siamese network research that focused on Natural Language Processing (NLP) task that we can find is the showcase of Long Short Term Memory (LSTM) and Siamese Network as a powerful language models to learn sentence similarity [21]. Using Manhattan Distance as an energy function, this architecture, which they call MaLSTM, exhibits superior performances compare to other methods existed before.

3. Methodologies

3.1. Data

In this paper, we adopted a publicly available argumentative relations dataset [14]. This dataset is the extended version of its predecessor [1]. The difference is in the amount of essays that are annotated, the previous version annotated 90 persuasive essays, while the last one 402 essays. In this corpus, argumentative relations are treated as a connected tree structure as seen in Fig. 1. This dataset proposed that there are 3 classes of argument components; major claims, claims and premises. Each premise might support or attack a claim or another premise. As what Fig. 1 suggests, this is a unidirectional graph, so if P2 supports P1, doesn't necessarily mean that P1 supports P2. Here are the example of argument relations:

Claim	Through cooperation, children can learn about interpersonal skills which are significant in the future life of all students
Premise 1	What we acquired from team work is not only how to achieve the same goal with others but more importantly, how to get along with others
Premise 2	During the process of cooperation, children can learn about how to listen to opinions of others, how to communicate with others, how to think comprehensively, and even how to compromise with other team members when conflicts occurred
Premise 3	All of these skills help them to get on well with other people and will benefit them for the whole life

The example above presented the support relations between Premise 1 and Claim, Premise 2 and Claim, and Premise 3 and Claim. Let's take a look at Premise 1 and Claim relation. Premise 1 gives a support to the Claim by saying that team work will help the children to acquire the ability to get along with others. This statement supports the main idea of the Claim, which is children can learn about interpersonal skills through cooperation.

However, there is no support relation between Premise 2 and Premise 1. Premise 2 does help to strengthen the main idea brought by the Claim because it also gives another example of benefits that children can gain from cooperation. But, premise 2 doesn't give any reasoning or example on why cooperation can help children to get along with others. Hence, no support relation exists between Premise 2 and Premise 1.

We attempted to follow the process of selecting the training and validation data elucidated previously[1] by omitting the argument relations generated by claims and major claims, and we obtain 820,182 pairs, which is comprised of 0.46% of supports and 99.54% of non-supports (table 1).

Table 1. Original data distribution of 402 essays dataset[14]

Supports	Non-supports	Total
3,794 (0.46%)	816,388 (99.54%)	820,182

It is an extremely imbalanced dataset. We attempted to take a closer look to the original experiment[1], we observe the distribution of their data which is also imbalanced. They managed to extract 6,330 pairs, of which 15.6% are support relations and 84.4% are non-support relations (table 2).

Table 2. Data distribution experiment, which comprised of 90 essays[1]

Supports	Non-supports	Total
989 (15.6%)	5,341 (84.4%)	6,330

The difference of the support relations data number is due to the amount of essays that we used, as explained above. We used the 402 essays dataset, while the state of the art experiment, used the 90 essays dataset[1]. Because of this extremely skewed dataset, we are afraid that it will cause an extreme overfitting problem. Hence, we do a simple randomize undersampling technique. In short, we randomize the non support relations data, and pick the data as much as the support relations data. We finally obtained a balanced dataset (table 3). This dataset is the one that we use in the experiment later on.

Table 3. Undersampled data distribution of 402 essays dataset

Supports	Non-supports	Total
3,794 (50.00%)	3,794 (50.00%)	7,588

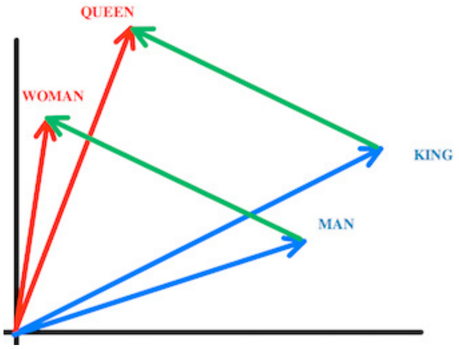

Fig. 2. An example of a word vector distribution. The distance between king and queen is the same with the distance between man and woman. From this vector relation, we can obtain the word queen by subtracting man vector from king vector and add woman vector

Once we obtain a balanced dataset, we need to manage the data so that it can be inserted into the neural network architecture. As the input of a Siamese Network, we need to create a pairing of data. Under the assumption of uni-directional graph, we create a pairing of data based on the argument relations graph and its direction. So if we take a look at Fig. 1, we create pairs of data based on the graph's edges. Claim 2 and P1 will be one pair of data, P2 and P1, P4 and P3, etc. Each pair of data will have its own label based on the argumentative relation it presented. Same treatment of pairing is also established for the non-support relations data. As for the labelling, we set 0 for support relations and 1 for the non-support relations under the assumption that if 2 sentence have a support relation, then they should have a smaller distance between each other.

3.2. Word Vector Representation

In order for text to be processed by Siamese Network computational graph, text needs to be represented by its vector form. We implemented a publicly available pre-trained word vector representation, GloVe[6]. We implemented GloVe as our word vector representation is because of its ability to preserve the relationships between words and its memory efficiency.

The most famous example of how word vector representation can preserve the word relations is *king - man + woman = queen* (see Fig. 2). This example shows that a vector *queen* can be obtained by subtracting vector *man* from vector *king* and adding vector *woman*. It means that word vector representation, in this case GloVe, is able to present the preserved vector relations. We adopted the 300d GloVe for this experiment, and it's treated as a fixed weight of the embedding layer of our Siamese Network architecture.

3.3. Recurrent Neural Network

Recurrent Neural Network (RNN) is a type of neural network that is able to process a variable-length sequential input[9]. The way RNN process a sequential data is by having a recurrent hidden units that will take every part of the sequence as their input. Every hidden unit in RNN is connected to the previous and the next timestep hidden unit. But, it is a challenging task to train RNNs to obtain long-term dependencies because of the gradients vanishing problem[22]. This makes the process of optimization using a gradient-based approach difficult. The way to solve this problem is by implementing a more sophisticated activation function, called gating function. One of the example is the Long Short Term Memory architecture, the earliest attempt to implement the gating function.

3.4. Long Short Term Memory

Long Short Term Memory (LSTM) was firstly introduced in 1997[8] as an alternative to solve the vanishing gradient problem. The reason why we choose LSTM as the network in this experiment is due to its empirically proven performance in handling variable-length sequential data. LSTM employed three gating mechanism, input gate, forget gate, and output gate. Input gate i_t regulates how much new information will be taken for the memory cell, while forget gate

f_t regulates the amount of the memory to be forgotten. The combination of the information from input gate and forget gate will become the new memory content, c_t. Output gate o_t modulates the amount of memory content exposure, which yields the output of the LSTM unit, h_t.

$$\begin{aligned}
i_t &= \sigma(W_i x_t + U_i h_{t-1} + b_i)\\
f_t &= \sigma(W_f x_t + U_f h_{t-1} + b_f)\\
\tilde{c}_t &= \tanh(W_c x_t + U_c h_{t-1} + b_c)\\
c_t &= i_t \odot \tilde{c}_t + f_t \odot c_{t-1}\\
o_t &= \sigma(W_o x_t + U_o h_{t-1} + b_o)\\
h_t &= o_t \odot \tanh(c_t)
\end{aligned} \tag{1}$$

In this paper, we attempted to analyze the best parameter of LSTM manually. We came to conclusion that LSTMs with 64 units present the most stable performance in this task, and the implementation of dropout[23] does help. We used 0.5 as the keep probability of the dropout mechanism.

3.5. Attention Mechanism

Consider an input which comes in form of sequence. For example, some words are sequentially combined to make one input. Every word in that sequence contributes to the meaning or representation of the entire sequence. But, it's often that not all of the words produce equal influential information. Hence, we need to assign a probability to every word to know how influential is it to the entire sequence.

$$\begin{aligned}
u_i &= \tanh(W_t h_i + b_t)\\
\alpha_i &= \frac{\exp(u_i^\top u_t)}{\sum_i \exp(u_i^\top u_t)}\\
\vec{v} &= \sum_i \alpha_i h_i
\end{aligned} \tag{2}$$

Essentially, attention mechanism measures the importance of a word or a part of a sequence through a softmax function and context vector, u_t. After computing the importance of each word, we obtain the sequence vector representation v as a weighted sum of all the words annotations based on the weight. In this paper, we observed the implementation of attention mechanism on top of the LSTM layer.

3.6. Siamese Network & Our Proposed Model

Siamese networks were firstly introduced to deal with a signature matching problem[17]. One siamese network originally consists of two identical networks with shared weights and accept two distict inputs which later be unified by an energy function. In short, Siamese networks compute the high level representations of the inputs first before the energy function computes some metrics from them (See Fig. 3).

Our proposed model did some changes on the original siamese network (See Fig. 4). The most notable is the LSTM layer as the high level feature extractors. So, in this experiment, LSTM will work as a features encoder. Based on our empirical analysis, cosine distance is the best energy function to represent the similarity of sentence semantic meaning.

Our model workflow can be seen in Fig. 4. Firstly, the model transforms the inputs, which comes in form of a pair of sentences, into its vector representation through the embedding lookup layer. As mentioned before, we took the benefit of GloVe as the fixed weight of the embedding layer. Embedding layer will then transform each word into its vector representation through a lookup process. Hence, the output will be a variable-length sequence of vector representation.

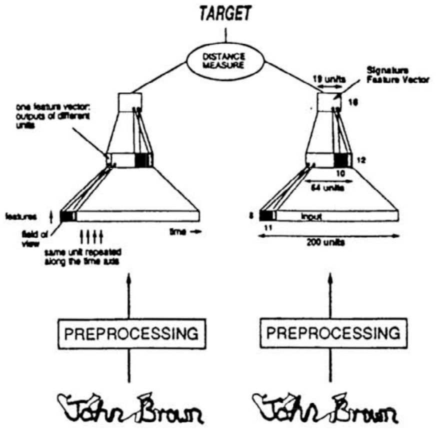

Fig. 3. Siamese network used in signature matching problem has shared weights, which means it will have an identical weights values in both sides of neural network [17].

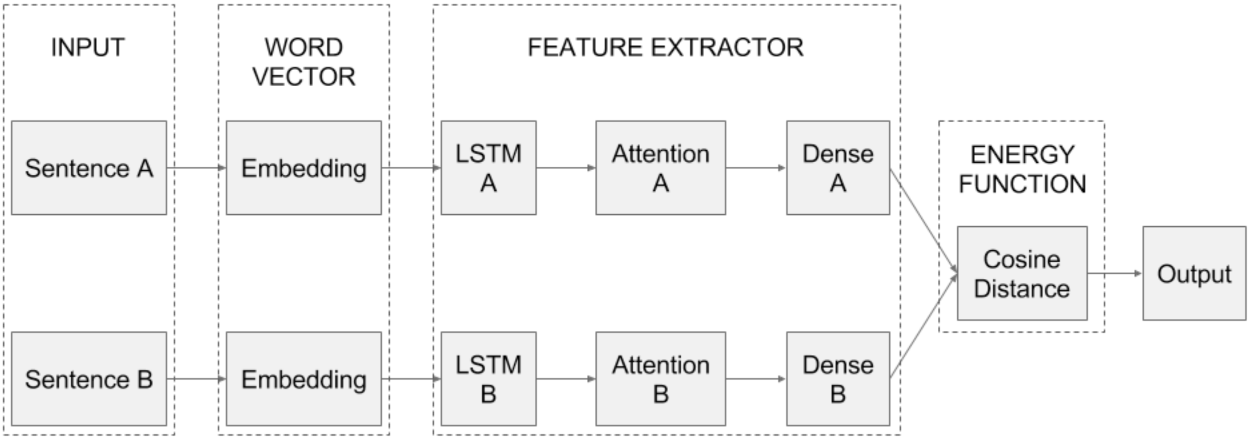

Fig. 4. Our proposed model for this task. LSTM siamese network has been used in sentence similarity detection task [21]. But, we implemented cosine distance instead of manhattan distance.

This output will then be inserted into the LSTM and Attention layer in both side. LSTM and Attention layer can be regarded as the feature extractor layer. Hence, the output of the aforemention layer is the high-level features of the input data. Both high level features will then be the input of the cosine distance layer input, which will calculate the difference between both features. We train the model using backpropagation-through-time under the contrastive loss function (See Eq. 3).

$$\begin{aligned}
\vec{v_A} &= \mathrm{LSTMAtt}_A(s_i) \\
\vec{v_B} &= \mathrm{LSTMAtt}_B(s_i) \\
D(\vec{v_A}, \vec{v_B}) &= \frac{\vec{v_A} \cdot \vec{v_B}}{\|\vec{v_A}\| \cdot \|\vec{v_B}\|} \\
L &= (1 - Y)\frac{1}{2}(D)^2 + (Y)\frac{1}{2}\{max(0, m - D)\}^2
\end{aligned} \tag{3}$$

Where, $\vec{v_A}$ and $\vec{v_B}$ are the output of the left and right LSTM+Attention layers respectively. $D(\vec{v_A}, \vec{v_B})$ is the cosine distance function, and L is the contrastive loss function. m is margin, where $m > 0$, and we set it to 1.

4. Results and Discussions

4.1. Results

For this experiment, we use Keras with TensorFlow backend. In this paper, we set the baseline which classify all pairs as non-supports. All of the models that used cosine distance produce a comparable F1 score, and higher accuracy, precision, and ROC-AUC score. On a balance dataset, we can take accuracy as a reliable performance metric as well. We can conclude that our models can outperform the baseline in majority of the performance metrics. In order to obtain a reliable result, we also implement stratified 10-fold cross validation. We choose the stratified method because we want to get an even distribution of dataset in every fold. The average of 10 fold results can be seen in Table. 4.

Table 4. Results of Siamese Network using LSTM and Attention with cosine distance. All of them adopted 300d GloVe as the embedding layer, with 64 as the batch size. LSTM denotes architectures that used LSTM as the only feature extractive layer, while LSTM+Att denotes architectures that used LSTM and Attention as the feature extractive layers

Model	Energy Function	Units	Accuracy	Precision	Recall	F1 Macro	ROC-AUC
Baseline	-	-	50.00 ± 0.00	50.00 ± 0.00	100.00 ± 0.00	66.67 ± 0.00	50.00 ± 0.00
Siamese (LSTM)	Manhattan	32	45.45 ± 2.64	45.3 ± 2.86	44.33 ± 7.46	44.59 ± 4.6	45.45 ± 2.64
Siamese (LSTM)	Manhattan	64	46.1 ± 2.64	46.17 ± 2.79	44.15 ± 9.3	44.58 ± 5.02	46.1 ± 2.64
Siamese (LSTM)	Manhattan	128	45.9 ± 1.83	45.74 ± 2.08	45.12 ± 7.82	45.18 ± 4.71	45.9 ± 1.83
Siamese (LSTM + Attention)	Manhattan	32	45.56 ± 1.73	45.11 ± 1.66	42.28 ± 12.71	42.78 ± 7.66	45.56 ± 1.73
Siamese (LSTM + Attention)	Manhattan	64	46.44 ± 2.17	45.28 ± 2.8	35.4 ± 14.09	38.47 ± 9.31	46.44 ± 2.17
Siamese (LSTM + Attention)	Manhattan	128	47.43 ± 1.37	43.79 ± 3.59	26.72 ± 23.7	29.66 ± 15.34	47.43 ± 1.37
Siamese (LSTM)	Euclidean	32	47.98 ± 2.02	45.88 ± 4.59	46.76 ± 33.2	41.55 ± 19.8	47.98 ± 2.02
Siamese (LSTM)	Euclidean	64	48.42 ± 1.36	46.81 ± 4.25	49.92 ± 35.11	42.93 ± 20.55	48.42 ± 1.36
Siamese (LSTM)	Euclidean	128	47.84 ± 1.9	46.59 ± 3.85	44.7 ± 35.29	38.95 ± 22.07	47.84 ± 1.9
Siamese (LSTM + Attention)	Euclidean	32	52.61 ± 5.51	51.97±4.18	95.87±8.72	66.98±0.67	52.61±5.51
Siamese (LSTM + Attention)	Euclidean	64	48.22 ± 1.75	47.89 ± 2.91	61.99 ± 33.85	49.46 ± 18.99	48.22 ± 1.75
Siamese (LSTM + Attention)	Euclidean	128	48.59 ± 1.88	46.74 ± 3.42	23.19 ± 9.22	30.08 ± 9.44	48.59 ± 1.88
Siamese (LSTM)	Cosine	32	52.61 ± 5.51	51.97±4.18	95.87±8.72	66.98±0.67	52.61±5.51
Siamese (LSTM)	Cosine	64	63.86 ± 2.24	60.56 ± 1.95	79.92 ± 3.3	68.86 ± 1.76	63.86 ± 2.24
Siamese (LSTM)	Cosine	128	63.65 ± 2.55	60.05 ± 2.34	82.74 ± 4.02	69.49 ± 1.23	63.65 ± 2.55
Siamese (LSTM + Attention)	Cosine	32	66.8 ± 1.46	68.31 ± 2.81	63.28 ± 5.32	65.51 ± 2.27	66.8 ± 1.46
Siamese (LSTM + Attention)	Cosine	64	67.33 ± 1.50	68.79 ± 3.12	64.07 ± 3.73	66.21 ± 1.27	67.33 ± 1.50
Siamese (LSTM + Attention)	Cosine	128	57.86 ± 7.04	55.47 ± 4.95	88.46 ± 7.72	67.85 ± 3.3	57.86 ± 7.04

The comparison of several results as seen in Table. 4 reveals that Siamese Network with 64 units of LSTM with Attention Mechanism using cosine distance achieves the best results. The model yields 67.33% of accuracy, 68.79% of precision, 64.07% of recall, 66.21% of f1 score and 67.33% of ROC-AUC score. We also attempted to analyze which class the neural network is struggling at predicting by producing the confusion matrix of the best model that we observed (see Table. 5). This confusion matrix is the average of the confusion matrices produced at every fold in the cross validation process.

Table 5. Confusion Matrix of Siamese network LSTM+attention with 64 units and cosine distance

	support	non-support
support	268	112
non-support	136	243

From Table. 5, we can conclude that the biggest mistake happened when the model tried to predict the non-support relations. There are 136 non-support pairs that are incorrectly classified as support relations. We do think it happened due to the inability of the model to classify reversed pairs. For example, if sentence A is supported by sentence B, the model will also predict that sentence B is supported by sentence A. Thus, this is not the problem of the data processing stage, but rather the model itself that exhibits a dissatisfying performances in distinguishing reversed relations. We also attempted to analyze this problem and found out that this occurs because of two reasons. First, the nature of siamese network itself, where it relies on the shared layer of feature extractor. Hence, the feature extracted from both sentence will always be the same even if switched and will yield an equal value of cosine distance. Second, the lack of energy function to capture a negative relation. All of the energy functions (Manhattan, Euclidean, and Cosine) only take the absolute difference between two data, and to the best of our knowledge, we cannot change to accommodate this problem.

To enable a deeper understanding, we also attempted to do several experiments using an imbalanced dataset. But, due to some ambiguity of dataset setting [1] from the prior experiment[1], we decided to not arbitrarily compare our results with the aforementioned experiment. We are still presenting the results of that experiment, but this should not be regarded as a comparative study between both experiments. These results are presented in Table. 6.

Table 6. Our proposed model performance in an imbalanced dataset. * prior research published result[1]

Model	Energy Function	Units	Accuracy	Precision	Recall	F1 Macro	ROC-AUC
SVM*	-	-	86.3	72.2	73.9	70.5	
Siamese (LSTM + Att)	Cosine	64	80.23 ± 1.82	41.93 ± 3.18	67.34 ± 4.41	51.61 ± 3.12	74.98 ± 2.30

Judging from Table. 6 presented ROC-AUC results, we believe that siamese network is able to perform well even in an imbalanced dataset. With 74.98 2.30% achieved by siamese network, proves that siamese network is able to adapt to imbalanced dataset as well.

4.2. Discussions

The results of the experiments enabled us to obtain several findings. The first one is the use of cosine distance as the energy function for this task can outperform other reasonable alternative of energy function (i.e. Manhattan, Euclidean). We do believe the reason behind it is the nature of cosine distance to capture a more representative similarity level in a multidimensional space vector data. The way cosine distance treat documents as vectors with its direction can capture the similarity better rather than just taking the real distance between two vectors like what Manhattan distance and Euclidean distance did.

The second finding is that smaller network can outperform bigger network in some cases. This is a common occurrence in deep learning due to underfitting problem. Larger network means more parameter to be trained. With a large amount of data, larger networks will be more favorable due to their ability to store this information. This happens because the networks will get a lot more data variations that allows their parameter to learn those variations. But in a smaller dataset, such as in this experiment, larger networks might not be able to achieve that. Larger networks will face a difficult time in generalizing while specializing their weights onto the data. Thus, smaller network can be more effective in this task.

The third finding is the effectiveness of attention mechanism. Attention mechanism is relatively able to give an increase to the performances of the model. As shown in Table. 4, the only decline of performances happened in the network with 128 units with attention mechanism. We believe that attention mechanism managed to boost the feature extraction process by giving a weighted access to each element of the sequence. This weighted access will then work as the weighted averaging agent that can calculate which element that the neural network needs to attend more.

The fourth finding is the implementation of siamese neural network in the task of identifying the support relations between two sentences. The result from this experiment can be used by future works as the performance comparison. Not to mention, this is one of the first attempt to use neural network in this particular task. The results presented by deep learning algorithm is quite satisfying. We do think that we can improve the performance by adding several model

[1] 1) The author mentioned that they were able to obtain 28,434 pairs of argument relations, while we cannot replicate this. 2) The author mentioned that only 4.6% of those pairs are the true instances but never explained what is and how to collect true instances. 3) The author also omit relations between claim and major claim, and managed to obtain 989 support relations, while we can only obtain 985 support relations

modifications, such as: an explicit memory to collect the extracted features. Unfortunately, in this paper we cannot present a comparative study between our proposed models with the state-of-the-art method[1], but we hope this can be done in the future.

5. Conclusion and Future Works

This work demonstrates the ability of deep learning to extract features from raw data and find similarity between two raw data. The implementation of LSTM Siamese Network with cosine similarity is quite unexplored, especially in argumentation mining task. To the best of our knowledge, this work is the one of the first experiment that uses deep learning to handle the task. In this paper, we are not attempting to compare our result with the prior research[1] due to the dataset difference. We do hope this work can be the benchmark of the future research in this task.

Since our approach is heavily rely on word vector representation, we do believe that we can improve the results by tweaking the choice of pre-trained word embedding. In the future, we'd like to try paragraph vector[24] due to its ability to present a sentence as a vector, rather than per word. The improvements in word embedding methods are also quite intriguing, since improvements in word embedding can provide a more comprehensive entity relationships[25].

Not only the word vector representation part, the advancement of memory network, such as Differentiable Neural Computer (DNC)[26]. We do believe the ability of explicit memory to save features can help the entire deep learning architecture in understanding the data and we will definitely do several experiments with it. We hope that we can do a lot more research on the implementation of deep learning in argumentation mining.

Acknowledgements

This research was fully funded by Bina Nusantara University Research Grant year 2017 with contract number 020A/VR.RTTO/IV/2017 and proposal id TI-04.

References

1. Stab, C., Gurevych, I.. Identifying argumentative discourse structures in persuasive essays. In: *EMNLP*. 2014, p. 46–56.
2. Desilia, Y., Utami, V.T., Arta, C., Suhartono, D.. An attempt to combine features in classifying argument components in persuasive essays. In: *17TH Workshop on Computational Models of Natural Argument (CMNA)*. 2017, .
3. Palau, R.M., Moens, M.F.. Argumentation mining: the detection, classification and structure of arguments in text. In: *Proceedings of the 12th international conference on artificial intelligence and law*. ACM; 2009, p. 98–107.
4. Adel, H., Roth, B., Schütze, H.. Comparing convolutional neural networks to traditional models for slot filling. *arXiv preprint arXiv:160305157* 2016;.
5. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.. Distributed representations of words and phrases and their compositionality. In: Burges, C.J.C., Bottou, L., Welling, M., Ghahramani, Z., Weinberger, K.Q., editors. *Advances in Neural Information Processing Systems 26*. Curran Associates, Inc.; 2013, p. 3111–3119.
6. Pennington, J., Socher, R., Manning, C.D.. Glove: Global vectors for word representation. In: *EMNLP*; vol. 14. 2014, p. 1532–1543.
7. Rumelhart, D.E., Hinton, G.E., Williams, R.J., et al. Learning representations by back-propagating errors. *Cognitive modeling* 1988;**5**(3):1.
8. Hochreiter, S., Schmidhuber, J.. Long short-term memory. *Neural computation* 1997;**9**(8):1735–1780.
9. Chung, J., Gülçehre, Ç., Cho, K., Bengio, Y.. Empirical evaluation of gated recurrent neural networks on sequence modeling. *Computing Research Repository* 2014;**abs/1412.3555**.
10. Kim, Y.. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:14085882* 2014;.
11. Habernal, I., Gurevych, I.. Which argument is more convincing? analyzing and predicting convincingness of web arguments using bidirectional lstm. In: *ACL (1)*. 2016, .
12. Stab, C., Gurevych, I.. Recognizing insufficiently supported arguments in argumentative essays. In: *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2017)*. Association for Computational Linguistics; 2017, p. 980–990.
13. Suhartono, D., Iskandar, A.A., Fanany, M.I., Manurung, R.. Utilizing word vector representation for classifying argument components in persuasive essays. In: *3rd International Conference on Science, Engineering, Built Environment, and Social Science (ICSEBS)*. 2016, .
14. Stab, C., Gurevych, I.. Parsing argumentation structures in persuasive essays. *Computational Linguistics* 2017;.
15. Egan, C., Siddharthan, A., Wyner, A.. Summarising the points made in online political debates. *ACL 2016* 2016;:134.
16. Marcu, D., Echihabi, A.. An unsupervised approach to recognizing discourse relations. In: *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*. Association for Computational Linguistics; 2002, p. 368–375.

17. Bromley, J., Guyon, I., LeCun, Y., Säckinger, E., Shah, R.. Signature verification using a" siamese" time delay neural network. In: *Advances in Neural Information Processing Systems*. 1994, p. 737–744.
18. Chopra, S., Hadsell, R., LeCun, Y.. Learning a similarity metric discriminatively, with application to face verification. In: *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*; vol. 1. IEEE; 2005, p. 539–546.
19. Yih, W.t., Toutanova, K., Platt, J.C., Meek, C.. Learning discriminative projections for text similarity measures. In: *Proceedings of the Fifteenth Conference on Computational Natural Language Learning*. Association for Computational Linguistics; 2011, p. 247–256.
20. Chen, K., Salman, A.. Extracting speaker-specific information with a regularized siamese deep network. In: *Advances in Neural Information Processing Systems*. 2011, p. 298–306.
21. Mueller, J., Thyagarajan, A.. Siamese recurrent architectures for learning sentence similarity. In: *AAAI*. 2016, p. 2786–2792.
22. Bengio, Y., Simard, P., Frasconi, P.. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks* 1994;**5**(2):157–166.
23. Srivastava, N., Hinton, G.E., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research* 2014;**15**(1):1929–1958.
24. Le, Q., Mikolov, T.. Distributed representations of sentences and documents. In: *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*. 2014, p. 1188–1196.
25. Li, Y., Xu, L., Tian, F., Jiang, L., Zhong, X., Chen, E.. Word embedding revisited: A new representation learning and explicit matrix factorization perspective. In: *IJCAI*. 2015, p. 3650–3656.
26. Graves, A., Wayne, G., Reynolds, M., Harley, T., Danihelka, I., Grabska-Barwińska, A., et al. Hybrid computing using a neural network with dynamic external memory. *Nature* 2016;**538**(7626):471–476.