APE: Argument Pair Extraction from Peer Review and Rebuttal via Multi-task Learning

Liying Cheng[*1,2] **Lidong Bing**[2] **Qian Yu**[†3] **Wei Lu**[1] **Luo Si**[2]
[1]Singapore University of Technology and Design
[2]DAMO Academy, Alibaba Group [3]The Chinese University of Hong Kong
{liying.cheng, l.bing, luo.si}@alibaba-inc.com
yuqian@se.cuhk.edu.hk luwei@sutd.edu.sg

Abstract

Peer review and rebuttal, with rich interactions and argumentative discussions in between, are naturally a good resource to mine arguments. However, few works study both of them simultaneously. In this paper, we introduce a new argument pair extraction (APE) task on peer review and rebuttal in order to study the contents, the structure and the connections between them. We prepare a challenging dataset that contains 4,764 fully annotated review-rebuttal passage pairs from an open review platform to facilitate the study of this task. To automatically detect argumentative propositions and extract argument pairs from this corpus, we cast it as the combination of a sequence labeling task and a text relation classification task. Thus, we propose a multi-task learning framework based on hierarchical LSTM networks. Extensive experiments and analysis demonstrate the effectiveness of our multi-task framework, and also show the challenges of the new task as well as motivate future research directions. [1]

1 Introduction

Argument mining is an important research field that attracts growing attention in recent years (Lawrence and Reed, 2019). It is applied in real-world applications such as legal documents (Mochales and Moens, 2011; Poudyal, 2015), online debates (Boltužić and Šnajder, 2015; Abbott et al., 2016), persuasive essays (Stab and Gurevych, 2014; Persing and Ng, 2016), etc. Most works in argument mining field focus on modeling arguments in monologues. Those focusing on the detection of agreement and disagreement in online interactions mainly fall in domains such as online debates and discussions (Chakrabarty et al., 2019).

Review:
[The authors introduce an extension of Continuous Ranked Probability Scores (CRPS) to the time-to-event setting termed Survival-CRPS for both right censored and interval-censored event data.]NON-ARGU ... [The claim that the proposed approach constitutes the first time a scoring rule other than maximum likelihood seems too strong, unnecessary and irrelevant to the value of the presented work.]REVIEW-1 [It is not clear how did the authors handle the irregularity (in time) of EHR encounters in the context of an RNN specification. Also, if the RNN specification considered is similar to Martinsson, 2016, why this wasn't considered as a competing model in the experiments?]REVIEW-2 ...

Rebuttal:
[Dear AnonReviewer3, we thank you for the time you spent reviewing and for the thoughtful comments.]NON-ARGU [You point out that our claim that the approach constitutes the first time a scoring rule other than maximum likelihood has been used seems too strong. ... However we have recently come across works such as adversarial time to event models, and therefore will remove the claim from the paper.]REPLY-1 [You also mention that it is unclear how irregularity (in time) of EHR encounters was handled by the RNN model. ... This approach naturally handles the irregularity of time between EHR encounters.]REPLY-2 ... [Thank you again for your time spent reviewing and constructive feedback on our work.]NON-ARGU

Table 1: An example of review-rebuttal passage pair. REVIEW-1 and REPLY-1 indicate the first argument in the review and its reply in the rebuttal.

Peer review is a widely-adopted evaluation process for scientific works (Kelly et al., 2014; Falkenberg and Soranno, 2018). As the number of submissions increases rapidly in recent years, how to evaluate the submissions effectively and efficiently becomes an attention-attracting challenge, and the study on peer review itself emerges as a new research topic (Xiong and Litman, 2011; Hua et al., 2019b). Nevertheless, the rebuttal part, which is often ignored, is an indispensable and interesting

[*]Liying Cheng is under the Joint Ph.D. Program between Alibaba and Singapore University of Technology and Design.
[†]Qian Yu was an intern at Alibaba.
[1]Our code and data are available at https://github.com/LiyingCheng95/ArgumentPairExtraction.

Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, pages 7000–7011,
November 16–20, 2020.

item in this scenario. Peer review and rebuttal, containing rich arguments, are a worth-studying domain for argument mining. In practice, a review and its rebuttal have strong interactions and relations, as a rebuttal usually contains coherent segments that argue with specific arguments in review. Compared to online debates and discussions, peer reviews and rebuttals on scientific works are a data source of rich structures and long passages for studying the argument pair extraction (APE). This study can also be applied to the study of online debates and discussions on how to respond to opponents' arguments.

Motivated by this, we aim to automatically extract the argument pairs from reviews and rebuttals by studying review-rebuttal pairs together in this work. In order to facilitate the study of this task, we introduce a newly-annotated dataset, named **R**eview-**R**ebuttal (RR) dataset. We create RR by first collecting all useful information related to the submissions from ICLR in recent years. In total, we select a corpus of 4,764 review-rebuttal passage pairs from over 22K reviews and author responses that are collected from the website. Then we annotate all these review and rebuttal passage pairs following a set of carefully defined annotation guidelines. Table 1 is an example of review-rebuttal passage pair, which shows the annotation result. The two passages of review and rebuttal are segmented into arguments and non-arguments. For arguments, indices are also labeled to show the alignment between review and rebuttal, i.e., the rebuttal argument that attacks the review argument.

With the prepared RR dataset, we focus on two subtasks in this work: (1) argument mining - detecting the arguments in each review/rebuttal by document segmentation. (2) sentence pairing - detecting if the sentences in a review argument form an argument pair with the sentences in a rebuttal argument. We cast the first subtask as a sentence-level sequence labeling task and the second subtask as a sentence-level binary classification task. As the two subtasks are mutually reinforcing each other, we then propose a multi-task learning framework based on the hierarchical bidirectional long short-term memory (LSTM) (Hochreiter and Schmidhuber, 1997) networks with a conditional random field (CRF) (Lafferty et al., 2001) layer (Lample et al., 2016). Our model allows learning better sentence representations for these two subtasks simultaneously. To better understand our model's capability, we evaluate the performance for each subtask separately as well as the overall argument pair extraction performance. Extensive experiments demonstrate the effectiveness of our model and challenges of our proposed task.

To summarize, our contributions include:

- We propose a new task of argument pair extraction from peer reviews and rebuttals. Meanwhile, a large corpus that facilitates this study is created.
- We propose a multi-task learning model that learns better sentence embeddings by coordinating the two subtasks.
- Extensive experiments and analysis demonstrate the effectiveness of our model, show the challenges of the new task, and also motivate future research directions.

2 Related Work

Argument Mining. There is an increasing number of works in the computational argumentation research field in recent years, such as argument mining (Shnarch et al., 2018; Trautmann et al., 2020), argument relation detection (Rocha et al., 2018; Hou and Jochim, 2017), argument quality assessment (Wachsmuth et al., 2017; Gleize et al., 2019; Toledo et al., 2019; Gretz et al., 2019), argument generation (Hua and Wang, 2018; Hua et al., 2019a; Schiller et al., 2020), etc. Stab and Gurevych (2014) and Persing and Ng (2016) both propose pipeline approaches to identify argumentative discourse structures in persuasive essays, which mainly includes two steps: extracting argument components and identifying relations. In terms of the task and the dataset, we intend to extract argument pairs from two passages simultaneously, while they focus on a single passage. In addition, we present a multi-task framework to tackle our proposed task instead of using a pipeline approach. Swanson et al. (2015) aim to extract arguments from a large corpus of posts from online forums. They frame their problem as two separate subtasks: argument extraction and argument facet similarity. Chakrabarty et al. (2019) focus on argument mining in online persuasive discussion forums based on the CMV dataset. Compared to both of their datasets, our dataset's contents are more academic and are strongly inclined to attack the other argument. Scale-wise, our dataset consists of 156K sentences, while Chakrabarty et al. (2019)'s dataset has 2,756 sentences.

Peer Reviews Mining. Previous works focus on studying peer reviews by introducing new datasets or understanding the structure and the content of peer reviews. Kang et al. (2018) present a dataset named PeerRead collected from venues including ACL, NIPS and ICLR. It consists of 14.7K paper drafts with their corresponding accept/reject decisions and 10.7K peer reviews for a subset of papers. They also propose several tasks based on their dataset, such as predicting whether a paper is accepted. Hua et al. (2019b) then present a partially annotated dataset named AMPERE collected from ICLR, UAI, ACL and NeurIPS. They use their dataset to detect the propositions and to classify them into different types for understanding the contents of peer reviews. Compared to the two works described above, our work has two main differences. First, instead of only looking at peer reviews, we propose a task to study the reviews and rebuttals jointly. Another difference lies in dataset: ours is fully annotated with 4,764 pairs of reviews and rebuttals, while only 400 reviews are annotated in AMPERE. As mentioned, few works touch on the rebuttal part, which is the other important element of the peer review process. Gao et al. (2019) propose a task to predict after-rebuttal scores from initial reviews and author responses. They collect their dataset from ACL, which includes around 4K reviews and 1.2K author responses. However, our argument pair extraction task focuses more on the internal structure and relations between reviews and rebuttals. In addition, the size of our dataset is much larger than theirs.

3 RR Dataset

We create the **R**eview-**R**ebuttal (RR) dataset to facilitate the study of argument pair extraction. To the best of our knowledge, this is the first dataset for argument mining between paired reviews and rebuttals. We first describe the process of data collection, and then present the annotation details.

Data Collection. To prepare our dataset, we first collect all useful information for ICLR 2013 - 2020 (except for 2015 that is unavailable) from `openreview.net`. The information mainly contains two parts: (1) information about the submissions, such as title, authors, year, track, acceptance decision, original submission separated by sections, etc.; (2) information about reviews and author responses, such as review passages, rebuttal passages, review scores, reviewer confidence, etc. Only review-rebuttal pairs are used in this work, while other data are reserved for further study on relevant tasks. In total, 22,127 reviews and rebuttals are collected. After excluding those reviews receiving no reply, we extract 4,764 review-rebuttal passage pairs for data annotation. For those reviews with multiple rounds of rebuttals, only the first rebuttal is selected.

Data Annotation. In this work, we concentrate on argument pair extraction from review-rebuttal passage pairs. In total, 5 professional data annotators are hired to annotate these 4,764 passage pairs based on the unified guidelines described below.

Firstly, for the argument mining part, the annotators segment reviews and rebuttals by labeling the arguments. In reviews, only sentences expressing non-positive comments are considered as arguments, e.g., review sentences that are related to specific suggestions, questions or challenges. In rebuttals, contents that are answering or explaining the specific review arguments are considered as arguments. All annotated arguments are semantically coherent segments. Sentences such as courtesy expressions or general conclusions are labeled as non-arguments. Note that the data are annotated on sentence granularity, i.e., an argument boundary is always a sentence boundary. Given the arguments of both passages, the second step is to pair an review argument with its corresponding rebuttal argument. We first assign indices to the arguments in reviews and then label the corresponding reply arguments in rebuttals with the same index. For example, in Table 1, {REVIEW-1, REPLY-1} and {REVIEW-2, REPLY-2} are two argument pairs.

We label 40,831 arguments in total, with 23,150 arguments from reviews and 17,681 arguments from rebuttals. We assess the annotation quality based on random sampling from the full dataset. 5% of the original review-rebuttal passage pairs (252 pairs) are checked. 39 out of 2,417 sampled arguments' labels are missed or have marginal error. Throughout the quality assessment, the annotation accuracy of the RR dataset reaches 98.4%.

We also classify the passages into *difficult* and *easy* based on the annotation difficulty when annotating the data. *Easy* is marked when the review and rebuttal passages have well-aligned arguments with clear structure. The annotators are able to identify the argument pairs without fully understanding the contents. For example, if the author cites the review by copying part of the review argument or

		RR
	# instances (i.e., review-rebuttal pairs)	4,764
Review	# sentences	99.8K
	# arguments	23.2K
	# argument sentences	58.5K
	Avg. # sentences per passage	21.0
	Avg. # sentences per argument	2.5
Rebuttal	# sentences	94.9K
	# arguments	17.7K
	# argument sentences	67.5K
	Avg. # sentences per passage	19.9
	Avg. # sentences per argument	3.8
Vocab.	Total vocab size	165K
	Review vocab size	103K
	Rebuttal vocab size	105K
	Avg. % vocab shared in each passage pair	16.6%
	Avg. % vocab shared in each argument pair	9.9%

Table 2: Overall statistics of RR dataset.

there are clear signals (i.e., line break, index, etc.) separating the arguments, it would be considered as *easy*. In total, 65.5% of the data (3,119 pairs) are classified as *easy*. *Difficult* is marked when the review and rebuttal passages do not have clear structure. The annotators have to understand the contents to identify the argument pairs. 29% of the data (1,383 instances) are classified as *difficult*. There are also 5.5% belonging to neither of the two classes, as there is no argument pair between them. This case usually happens when the rebuttal part only contains a few sentences by a general reply, such as "Thanks for your review. We will use your suggestions to improve our work." or "Thanks, your comments were helpful and we've taken them into account in the updated paper.". Note that in our dataset, we keep the original structure information in the plain-text, such as line breaks and indents.

Data Analysis. The overall statistics are shown in Table 2. Although the rebuttal passages are generally shorter than the review passages (i.e., more sentences in each review passage on average), the rebuttal arguments are relatively longer than the review arguments. Additionally, a larger portion of sentences are arguments in rebuttals than in reviews. In the rebuttal passages, authors tend to only focus on replying reviewer's arguments by explaining their points clearly. While reviewers are more likely to provide summaries and compliments about the submissions apart from questions and suggestions, which are not considered as arguments in this work. As shown in the vocabulary part from Table 2, the average percentage of vocabulary shared between each review and rebuttal argument pair is only 9.9%, which is much lower than the ratio across entire passages (16.6%). This indicates that conducting argument pair extraction on our dataset is very challenging since there are fewer shared words in a pair.

4 Task Description

In this work, we intend to automatically detect the argument pairs from each passage pair of review and rebuttal. Specifically, we first perform argument mining on the two passages, a review passage of m sentences $R_1 = [s_1^1, s_1^2, \cdots, s_1^m]$ and a rebuttal passage of n sentences $R_2 = [s_2^1, s_2^2, \cdots, s_2^n]$. Each mined argument consists of one or more sentences from the passage with no overlapping with other arguments, such as REVIEW-1, REVIEW-2, REPLY-1, and REPLY-2 in Table 1. Then we extract the argument pairs by matching individual review arguments with the rebuttal arguments, such as {REVIEW-1, REPLY-1} and {REVIEW-2, REPLY-2}.

Data Processing. Based on the annotation, we further process our dataset by segmentation and re-labeling in order to fit the need of downstream models. Accordingly, each segment in the passage (for both argument and non-argument) is further segmented into sentences. Each sentence is then assigned a label according to the standard IOBES tagging scheme (Ramshaw, 1995; Ratinov and Roth, 2009). For example, in Table 1, each sentence in NON-ARGU is labeled as O; REVIEW-1 is labeled as S since there is only one sentence in that argument; the two sentences in REVIEW-2 are labeled as {B, E} in sequence. In addition, we assign a binary label for each pair of review sentence and rebuttal sentence to indicate if the two sentences in this pair are aligned. Specifically, the binary label is 1 only when: 1) both two sentences are from arguments; 2) the rebuttal sentence is replying the argument where the review sentence originates. Otherwise, 0 will be assigned.

5 Multi-task Learning based Extraction

As shown in Figure 1, we propose a multi-task learning framework with hierarchical LSTM networks, named MT-H-LSTM-CRF, to tackle the two subtasks, namely, argument mining and sentence pairing. The red dotted box on the left shows our sentence encoder which uses the pre-trained

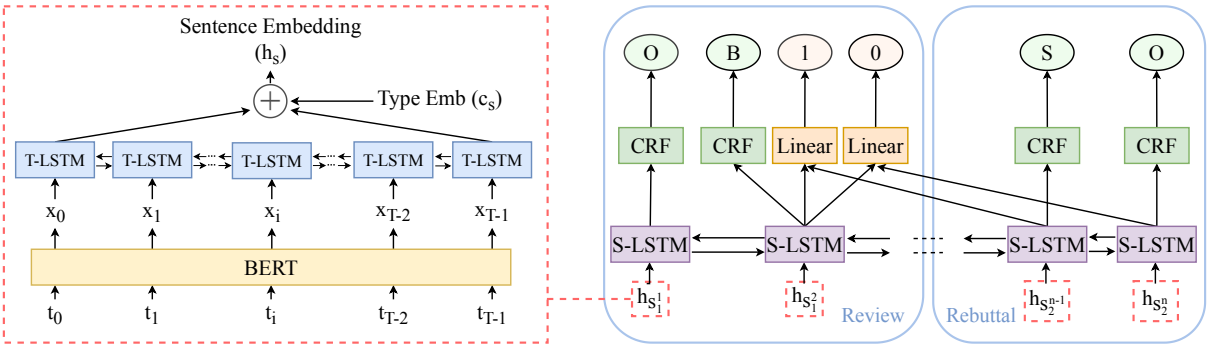

Figure 1: Overview of our MT-H-LSTM-CRF model architecture. The sentence encoder (in the red dotted box on the left) shows the process of obtaining sentence embeddings from pre-trained BERT token embeddings with T-LSTM. The multi-task learning based framework (on the right) shows the process of generating labels for both subtasks with the shared S-LSTM.

BERT (Devlin et al., 2019) token embeddings as the input of a token-level biLSTM (T-LSTM) to get sentence embeddings. Then, the encoded sentence embeddings are input to the second layer of LSTM on the sentence level (S-LSTM) to encode the sequence information of the passages. Finally, two types of labels, i.e., IOBES and 1/0, are predicted simultaneously from the shared features generated from the S-LSTM. Note that the aforementioned procedure is of multi-task training. During the inference, the trained multi-task model will be decoupled into two sub-modules to perform the two subtasks in a pipeline manner to extract the final argument pairs.

5.1 Sentence Encoder

We first introduce how to get the sentence embeddings as the input for our main model. As shown in the left part of Figure 1, the input is a sentence with a list of T tokens $s = [t_0, t_1, \cdots, t_{T-1}]$. We adopt bert-as-service (Xiao, 2018) as a tool to obtain the embeddings for all tokens $[x_0, x_1, \cdots, x_{T-1}]$ in the sentence. The token embeddings are then fed into a token-level biLSTM layer (T-LSTM), where the last hidden states from both directions are concatenated as the sentence embedding. In order to distinguish the sentence type (i.e., review or reply), we concatenate the current sentence embedding with a trainable type embedding c_s that is randomly initialized, to obtain the final sentence embedding h_s for the main model on the right.

5.2 biLSTM-CRF

After we obtain the sentence embeddings with T-LSTM as the input, we adopt a biLSTM-CRF structure to perform the sentence-level IOBES tagging for mining arguments. As shown in the right part of Figure 1, after the S-LSTM encodes the sequence information of a sentence sequence, the CRF layer takes the feature of each position to predict tags.

Given a sequence of review-rebuttal sentences $\mathbf{s} = \{s_1^1, s_1^2, \cdots, s_1^m, s_2^1, s_2^2, \cdots, s_2^n\}$, which is the concatenation of R_1 and R_2, our task is to predict a label sequence $\mathbf{y}$ where each element y_i belongs to the label set $\{\text{B}, \text{I}, \text{E}, \text{S}, \text{O}\}$. The probability of a label sequence $\mathbf{y}$ given $\mathbf{s}$ is defined as:

$$p(\mathbf{y}|\mathbf{s}) = \frac{\exp(score(\mathbf{s}, \mathbf{y}))}{\sum_{\mathbf{y}} \exp(score(\mathbf{s}, \mathbf{y}))},$$

where the $score(\mathbf{s}, \mathbf{y})$ is usually defined as a linear function in traditional CRF models. In the neural CRF model, the score can be obtained from the neural network encoders such as biLSTM networks. As shown in the following equation, it is calculated by the sum of transition scores along the label sequence $\mathbf{y}$ and the scores from the neural networks:

$$score(\mathbf{s}, \mathbf{y}) = \sum_{i=0}^{n} A_{y_i, y_{i+1}} + \sum_{i=1}^{n} F_{\theta_1}(\mathbf{s}, y_i),$$

where $A_{y_i, y_{i+1}}$ represents the transition parameters between two labels, and $F_{\theta_1}(\mathbf{s}, y_i)$ indicates the score of y_i obtained from the neural network encoder parameterized by θ_1. y_0 and y_{n+1} represent the "START" and "END" labels, respectively. We aim to minimize the negative log-likelihood for our dataset $\mathcal{D}_1$:

$$\mathcal{L}_1(A, \theta_1|\mathcal{D}_1) = -\sum_{(\mathbf{s},\mathbf{y})\in\mathcal{D}_1} \log p(\mathbf{y}|\mathbf{s}).$$

The gradients with respect to the parameters can be calculated efficiently through the forward-backward algorithm in the CRF layer and back propagation in the neural networks.

During decoding, we retrieve the label sequence through Viterbi algorithm over all the possible label sequences:

$$\mathbf{y}^* = \arg\max_{\mathbf{y}\in\mathcal{Y}} p(\mathbf{y}|\mathbf{s}).$$

5.3 biLSTM-linear

In order to predict the pairing relation among arguments, our framework incorporates a sentence-level binary classification task. The classifier predicts that if two sentences belong to the same pair of argument, we then utilize the predictions to infer the pairing information between two augments.

Specifically, we sum up the S-LSTM outputs of the review sentence and the rebuttal sentence in a pair as the pair feature:

$$h^{i,j}_{pair} = h_{s^i_1} + h_{s^j_2}, \forall s^i_1 \in R_1, s^j_2 \in R_2.$$

Then $h^{i,j}_{pair}$ is fed into linear layers to predict the binary label $z^{i,j}_{pair} \in \{0, 1\}$. The negative log-likelihood for our dataset $\mathcal{D}_2$ parameterized by θ_2 is minimized as follows:

$$\mathcal{L}_2(\theta_2|\mathcal{D}_2) = -\sum_{(h_{pair},z)\in\mathcal{D}_2} \Big(z \log p(z=1|h_{pair}) + (1-z)\log p(z=0|h_{pair})\Big).$$

During decoding, as shown in Figure 1, each binary tag is predicted as:

$$z^* = \arg\max_{z\in\{0,1\}} p(z|h_{pair}).$$

Here the sentence representations from S-LSTM are shared by biLSTM-linear and biLSTM-CRF, which coordinate the two subtasks to achieve the best performance on argument pair extraction.

Negative Sampling. Obviously, according to the data processing step in Section 4, the classification dataset with binary labels is unbalanced. To ease the problem, we adopt negative sampling techniques (Mikolov et al., 2013). During training, for each review argument sentence, we randomly select k non-aligned rebuttal argument sentences as negative samples. These k negative samples together with all positive pairs form our dataset $\mathcal{D}_2$.

5.4 Multi-task Training

To conduct the multi-task learning, we simply sum up the losses of the two subtasks:

$$\mathcal{L} = w_1\mathcal{L}_1(A, \theta_1|\mathcal{D}_1) + w_2\mathcal{L}_2(\theta_2|\mathcal{D}_2),$$

where w_1 and w_2 are weights for each loss accordingly. Note that in the training phase, we do not check the performance of individual subtasks, as we select the best model according to the overall argument pair extraction result on the development set. During the inference, we first mine the arguments with the sequence labeling module and then feed the labeled results of argument mining into the binary classifier to conduct sentence pairing.

6 Experiments

6.1 Data Preparation

We first split our RR dataset on review-rebuttal passage-pair level randomly by a ratio of 8:1:1 for training, development and testing, namely **RR-passage**. In RR-passage dataset, all argument pairs from the same passage pair are put into only one of the training, development and testing sets. However, different review-rebuttal passage pairs of the same submission could be put into different sets.

Since different reviewers may discuss similar issues for one submission, different review-rebuttal passage pairs of the same submission may share similar context information. To alleviate this effect, we also prepare another dataset version split on the submission level, namely **RR-submission**. It also follows the ratio of 8:1:1 for training, development and testing tests. In RR-submission, multiple review-rebuttal passage pairs of the same submission are in the same set.

6.2 Experimental Settings

We implement our multi-task framework in PyTorch. All models are run with V100 GPU. The dimension of pre-trained BERT sentence embeddings is 768 by default. The dimension of type embedding is set as 20. Number of T-LSTM layers is set as 1. Number of S-LSTM layers is set as 2. Number of linear layers for binary classification task is 3. We randomly select 5 negative samples for the classification task during training in main experiments. We use Adam (Kingma and Ba, 2014) with an initial learning rate of 0.01 and update parameters with a batch size of 10. The weights shared between two losses w_1 and w_2 are both set as 0.5. The training phase is stopped when detecting the convergence on the validation set. More details of parameter setting, such as the number of LSTM/linear layers, the weights of losses, and the comparison between sum and concatenation for sentence pair representation, can be found in Appendix A. Note that in this paper, the parameters are mainly tuned based on RR-passage.

Models	Argument Mining			Sentence Pairing			Argument Pair Extraction		
	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
PL-H-LSTM-CRF	**73.10**	67.65	70.27	62.52	**75.32**	**68.32**	21.24	19.30	20.23
MT-LSTM-CRF	61.02	54.72	57.70	72.46	62.09	66.87	22.09	18.95	20.40
Hybrid-MT-H-LSTM-CRF	70.80	68.38	69.57	56.19	13.23	21.41	23.65	22.30	22.95
MT-H-LSTM-CRF (Ours)	71.85	**71.01**	**71.43**	**72.64**	58.05	64.53	**30.08**	**29.55**	**29.81**

Table 3: Main results on **RR-passage** dataset.

6.3 Results on RR-passage Dataset

6.3.1 Results of Argument Pair Extraction

In the testing stage, our trained multi-task framework is applied as two modules to conduct the subtasks of argument mining and sentence pairing sequentially. Thus, we evaluate our model in three aspects: argument mining, sentence pairing, and argument pair extraction. After the argument mining step, we evaluate the mining result by checking the correctness of each argument span consisting of one or a few sentences labeled with IOBES. The sentence pairing step takes the mining results to predict the pairing relation of two predicted argument spans. This step first conducts prediction on sentence pairs, and then applies the sentence pairing results to infer the argument pairing relation. More details on how to determine the argument pairing relation can be found in Appendix B. After this step, we obtain the final results of argument pair extraction and check their correctness against the gold labeled argument pairs. We also evaluate the performance of binary classification for the sentence pairing step on the sentence pair data, as described in the data processing step in Section 4.

Main Results. Table 3 shows the performance on both subtasks as well as the overall extraction performance on RR-passage dataset, where we compare our proposed multi-task model (**MT-H-LSTM-CRF**) with several strong baselines. The first model is a pipeline approach, and the others are all multi-task learning based. As mentioned before, for the multi-task learning models, we select the best model according to the final argument pair extraction performance on the development set.

First, **PL-H-LSTM-CRF** is a pipeline approach that trains argument mining and sentence pairing modules independently and then pipes them together to extract argument pairs. Although **PL-H-LSTM-CRF** achieves competitive performance on argument mining task and the highest F1 score on sentence pairing task, its overall extraction performance is much worse than our multi-task model **MT-H-LSTM-CRF**. The main reason is that the sentence embeddings in the pipeline approach are not shared. Thus, although these two subtasks can be well learned separately, they are not trained to collaborate with each other. Its lower overall performance shows the necessity of multi-task training for precise argument pair extraction.

In contrast, our multi-task model shares sentence embeddings by the hierarchical LSTM design. **MT-LSTM-CRF** is a baseline without the hierarchical LSTM design. Specifically, T-LSTM is removed from our framework. Compared with **MT-H-LSTM-CRF**, **MT-LSTM-CRF** achieves much worse performance on argument mining task and overall extraction evaluation, but competitive performance on sentence pairing task. It shows the importance of the hierarchical design on the argument mining task because T-LSTM learns the semantics of individual sentences on token granularity, which is important for predicting if a sentence is argumentative. Compared with the pipeline method, i.e., **PL-H-LSTM-CRF**, **MT-LSTM-CRF** still achieves slightly better overall extraction performance, which shows the shared S-LSTM layer in the multi-task learning can help to coordinate the training of two subtasks.

To further verify the effectiveness of our multi-task learning through the hierarchical LSTM design, we investigate another baseline, namely, **Hybrid-MT-H-LSTM-CRF**, which uses the sentence embeddings from T-LSTM as the features for the sentence pairing classification and uses the embeddings from S-LSTM for labeling argument sentences. Here, the performance of the sentence pairing task is sacrificed for better overall extraction result, as the best model is selected based on the F1 score of argument pair extraction. With no incorporation of the context information captured by S-LSTM, the sentence pairing task performance drops significantly, which leads to poorer overall extraction performance compared to **MT-H-**

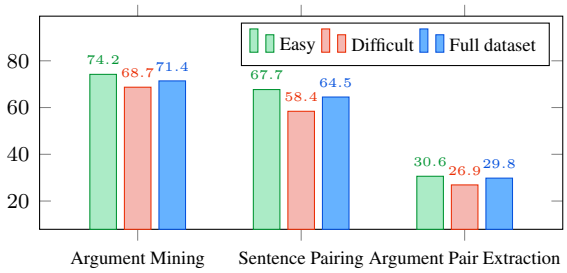

Figure 2: Breakdown F1 results by difficulty.

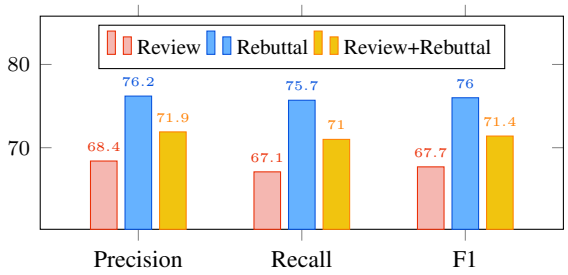

Figure 3: Comparison of argument mining results between review and rebuttal.

Models	Precision	Recall	F1
LSTM-CNN	51.42	41.01	47.29
LSTM-CRF	55.74	43.20	50.33
LSTM-CRF-type	62.98	56.24	59.42
H-LSTM-CRF (Ours)	**73.10**	**67.65**	**70.27**

Table 4: Performance on argument mining task.

# NS	P:N	Precision	Recall	F1
4Argu	1:1.2	29.87	28.35	29.09
5Argu	1:1.4	30.08	**29.55**	**29.81**
6Argu	1:1.6	**30.59**	28.80	29.67
No sampling	1:15.1	29.34	27.40	28.34

Table 5: Argument pair extraction results under different negative samples (NS).

LSTM-CRF. This observation again verifies the importance of sharing ample information between two subtasks with the hierarchical LSTMs design.

Comparison among different difficulty levels. We further evaluate the performance of different difficulty levels. As mentioned earlier, difficulty levels are annotated based on whether the annotators are able to extract the argument pairs by looking at the structure but without fully understanding the specific contents. It is clear to see from Figure 2 that the performance for *easy* instances is obviously better on overall argument pair extraction as well as two individual subtasks. This indicates that the structure information of review-rebuttal passages plays an important role in both argument mining and sentence pairing.

6.3.2 Argument Mining Subtask Evaluation

Here we examine the performance of different models on the argument mining task. The results are shown in Table 4. For MT-H-LSTM-CRF, we downgrade it to **H-LSTM-CRF** by only training the argument mining module, which is the same as the first step of **PL-H-LSTM-CRF**. The first two, i.e., **LSTM-CNN** (Chiu and Nichols, 2016) and **LSTM-CRF**, are classical sequential labeling models, which take the sentence embeddings from BERT as input of the LSTM layer. **LSTM-CRF-type** takes the concatenation of sentence embeddings and type (i.e., review or rebuttal) embeddings as input, which is the same as the first step of **MT-LSTM-CRF**. **LSTM-CRF-type** performs much better than **LSTM-CRF**. This indicates the usefulness of type embeddings. **H-LSTM-CRF** achieves much better results, which is consistent with our conclusion on the importance of our hierarchical LSTM design in the previous subsection. Note that as shown in Table 3, the results of **MT-LSTM-CRF** for the first subtask are a bit lower than those of **LSTM-CRF-type** here, which is mainly because the multi-task training negatively affects the former's performance. However, when we compare the results of **MT-H-LSTM-CRF** for the first subtask in Table 3 with the performance here of **H-LSTM-CRF**, the former even performs better, which shows that after adding the T-LSTM layer, our multi-task model's capability for argument mining is indeed benefited.

Figure 3 shows the argument mining results on reviews and rebuttals respectively. It is clear that the model performs significantly better on the rebuttal passages than on the review passages. This suggests that rebuttals have more organized and clearer structure than reviews. Indeed, in the rebuttal phase, authors reply reviewer's suggestions and questions very carefully to make the points clear, sometimes by citing the review arguments. Thus, the structure and the format of rebuttals are relatively fixed, while reviewers have more flexibility in the style and the structure when writing reviews.

6.3.3 Data Sampling Strategies

For training the sentence pairing classifier, we investigate the effect of data sampling on the matching performance. Specifically, we compare the

Models	Argument Mining			Sentence Pairing			Argument Pair Extraction		
	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
PL-H-LSTM-CRF	67.63	68.51	68.06	61.01	55.49	58.12	19.86	19.94	19.90
MT-LSTM-CRF	62.38	54.29	58.05	**72.63**	**59.79**	**65.59**	23.28	18.80	20.80
Hybrid-MT-H-LSTM-CRF	69.39	69.95	69.67	49.60	13.64	21.40	21.95	22.12	22.04
MT-H-LSTM-CRF (Ours)	**70.09**	**70.14**	**70.12**	71.47	52.61	60.61	**26.69**	**26.24**	**26.46**

Table 6: Main results on **RR-submission** dataset.

performance of using different numbers (i.e., k) of negative samples for each review argument sentence. Table 5 shows the results for argument pair extraction, as well as the ratio of positive samples to negative samples (P:N). When training with all non-aligned sentence pairs as negative samples, i.e., **No sampling**, the ratio of aligned pairs to non-aligned pairs is 1:15.1, which is obviously unbalanced leading to the poor results. We then select only 4/5/6 non-aligned rebuttal argument sentences randomly as negative samples for each review argument sentence, i.e., **4Argu**, **5Argu** and **6Argu**. Generally speaking, the ratio is more balanced, and the overall extraction performance improves, and of course, the training is more efficient. Especially, when 5 non-aligned argument pairs are selected (**5Argu**), our model achieves the best F1 score (29.81) for argument pair extraction.

6.4 Results on RR-submission Dataset

We also conduct the major experiments on RR-submission dataset with the same settings as used on RR-passage. The results on argument mining, sentence pairing and argument pair extraction are shown in Table 6. We observe that our proposed **MT-H-LSTM-CRF** consistently outperforms the baseline models. However, it performs slightly worse on RR-submission than on RR-passage, plausibly because there is no context information (i.e., background knowledge from original submissions) shared between different passage pairs.

7 Conclusions and Future Work

In this paper, we introduce a new task of extracting argument pairs from review and rebuttal passages, which explores a new domain for the argument mining research field. A new large-scale and challenging dataset RR is collected and fully annotated to facilitate the study of the proposed task. We then propose a multi-task learning approach based on hierarchical LSTM networks to work towards this problem. In the future, we will explore the latent information between peer reviews and author responses to improve argument pair extraction. We will also explore related useful research tasks using extra collected information related to scientific work submissions in RR.

Acknowledgements

We would like to thank the anonymous reviewers for their constructive comments on this work. We would also like to thank Zhanming Jie and Tianyu Wu for the fruitful discussion.

References

Rob Abbott, Brian Ecker, Pranav Anand, and Marilyn Walker. 2016. Internet argument corpus 2.0: An sql schema for dialogic social media and the corpora to go with it. In *Proceedings of LREC*.

Filip Boltužić and Jan Šnajder. 2015. Identifying prominent arguments in online debates using semantic textual similarity. In *Proceedings of the 2nd Workshop on Argumentation Mining*.

Tuhin Chakrabarty, Christopher Hidey, Smaranda Muresan, Kathy McKeown, and Alyssa Hwang. 2019. AMPERSAND: Argument mining for PERSuAsive oNline discussions. In *Proceedings of EMNLP-IJCNLP*.

Jason PC Chiu and Eric Nichols. 2016. Named entity recognition with bidirectional lstm-cnns. *TACL*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL*.

Laura J Falkenberg and Patricia A Soranno. 2018. Reviewing reviews: An evaluation of peer reviews of journal article submissions. *Limnology and Oceanography Bulletin*.

Yang Gao, Steffen Eger, Ilia Kuznetsov, Iryna Gurevych, and Yusuke Miyao. 2019. Does my rebuttal matter? insights from a major nlp conference. In *Proceedings of NAACL*.

Martin Gleize, Eyal Shnarch, Leshem Choshen, Lena Dankin, Guy Moshkowich, Ranit Aharonov, and

Noam Slonim. 2019. Are you convinced? choosing the more convincing evidence with a siamese network. In *Proceedings of ACL*.

Shai Gretz, Roni Friedman, Edo Cohen-Karlik, Assaf Toledo, Dan Lahav, Ranit Aharonov, and Noam Slonim. 2019. A large-scale dataset for argument quality ranking: Construction and analysis. In *Proceedings of AAAI*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Yufang Hou and Charles Jochim. 2017. Argument relation classification using a joint inference model. In *Proceedings of the 4th Workshop on Argument Mining*.

Xinyu Hua, Zhe Hu, and Lu Wang. 2019a. Argument generation with retrieval, planning, and realization. In *Proceedings of ACL*.

Xinyu Hua, Mitko Nikolov, Nikhil Badugu, and Lu Wang. 2019b. Argument mining for understanding peer reviews. In *Proceedings of NAACL*.

Xinyu Hua and Lu Wang. 2018. Neural argument generation augmented with externally retrieved evidence. In *Proceedings of ACL*.

Dongyeop Kang, Waleed Ammar, Bhavana Dalvi, Madeleine van Zuylen, Sebastian Kohlmeier, Eduard Hovy, and Roy Schwartz. 2018. A dataset of peer reviews (peerread): Collection, insights and nlp applications. In *Proceedings of NAACL*.

Jacalyn Kelly, Tara Sadeghieh, and Khosrow Adeli. 2014. Peer review in scientific publications: benefits, critiques, & a survival guide. *EJIFCC*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. Technical report.

John Lafferty, Andrew McCallum, and Fernando CN Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of ICML*.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of NAACL*.

John Lawrence and Chris Reed. 2019. Argument mining: A survey. *Computational Linguistics*, 1(1).

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Proceedings of NIPS*.

Raquel Mochales and Marie-Francine Moens. 2011. Argumentation mining. *Artificial Intelligence and Law*.

Isaac Persing and Vincent Ng. 2016. End-to-end argumentation mining in student essays. In *Proceedings of NAACL*.

Prakash Poudyal. 2015. A machine learning approach to argument mining in legal documents. In *AI Approaches to the Complexity of Legal Systems*. Springer.

LA Ramshaw. 1995. Text chunking using transformation-based learning. In *Proceedings of Third Workshop on Very Large Corpora*.

Lev Ratinov and Dan Roth. 2009. Design challenges and misconceptions in named entity recognition. In *Proceedings of CoNLL*.

Gil Rocha, Christian Stab, Henrique Lopes Cardoso, and Iryna Gurevych. 2018. Cross-lingual argumentative relation identification: from english to portuguese. In *Proceedings of the 5th Workshop on Argument Mining*.

Benjamin Schiller, Johannes Daxenberger, and Iryna Gurevych. 2020. Aspect-controlled neural argument generation. *CoRR*.

Eyal Shnarch, Carlos Alzate, Lena Dankin, Martin Gleize, and Noam Slonim. 2018. Will it blend? blending weak and strong labeled data in a neural network for argumentation mining. In *Proceedings of ACL*.

Christian Stab and Iryna Gurevych. 2014. Identifying argumentative discourse structures in persuasive essays. In *Proceedings of EMNLP*.

Reid Swanson, Brian Ecker, and Marilyn Walker. 2015. Argument mining: Extracting arguments from online dialogue. In *Proceedings of SIGDIAL*.

Assaf Toledo, Shai Gretz, Edo Cohen-Karlik, Roni Friedman, Elad Venezian, Dan Lahav, Michal Jacovi, Ranit Aharonov, and Noam Slonim. 2019. Automatic argument quality assessment – new datasets and methods. In *Proceedings of EMNLP*.

Dietrich Trautmann, Johannes Daxenberger, Christian Stab, Hinrich Schütze, and Iryna Gurevych. 2020. Fine-grained argument unit recognition and classification. In *Proceedings of AAAI*.

Henning Wachsmuth, Nona Naderi, Ivan Habernal, Yufang Hou, Graeme Hirst, Iryna Gurevych, and Benno Stein. 2017. Argumentation quality assessment: Theory vs. practice. In *Proceedings of ACL*.

Han Xiao. 2018. bert-as-service. https://github.com/hanxiao/bert-as-service.

Wenting Xiong and Diane Litman. 2011. Automatically predicting peer-review helpfulness. In *Proceedings of ACL*.

A Additional Experiments

Effect of Number of T-LSTM Layers. We compare the performance of argument pair extraction and two subtasks when using different numbers of T-LSTM layers (i.e., L_T). L_S denotes the number of S-LSTM layers. In Table 7, **MT-LSTM-CRF** is the model when T-LSTM is not adopted (i.e., the number of T-LSTM layers is 0). With the incorporation of the T-LSTM, **MT-H-LSTM-CRF** outperforms **MT-LSTM-CRF**. As we can see, **MT-H-LSTM-CRF**(L_T**=1,**L_S**=1**) performs better than **MT-H-LSTM-CRF**(L_T**=2,**L_S**=1**). This suggests that long-distance dependency among tokens does not help for sentence embedding learning.

Effect of Number of S-LSTM Layers. We compare the performance when using different numbers of S-LSTM layers (i.e., L_S). As we can see from Table 8, **MT-H-LSTM-CRF**(L_T**=1,**L_S**=2**) outperforms the other two models for argument mining subtask and argument pair extraction task. This again shows the importance of context information in the review and rebuttal passages. However, increasing the number of layers (e.g., L_S=3) does not gain further performance improvements for **MT-H-LSTM-CRF**. This implies that the third-order context information in the passages does not play an important role in extracting argument pairs.

Effect of Number of Linear Layers. We also evaluate the performance when using various numbers of linear layers for sentence pairing task. The results are shown in Table 9. Here, we fix the number of S-LSTM layers and the number of T-LSTM layers both at 1. When the number of linear layers is 2, **MT-H-LSTM-CRF**(L_T**=1,**L_S**=1**) performs the best on sentence pairing task. However, the performance reaches the peak when we increase the number of linear layers to 3 and drops significantly when we further increase the number of linear layers to 4. This suggests too few or too many parameters may harm the model performance.

Effect of Weight Ratio between two Losses. In order to evaluate the effect of the weights for two subtasks in our proposed multi-task model, we compare the performance when assigning different weights for both two subtasks. The results are shown in Table 10. For the argument pair extraction, **MT-H-LSTM-CRF** performs the best when equal weight is assigned across two subtasks. The performance of both argument mining subtask and argument pair extraction task drops consistently when more weights are assigned for sentence pairing subtask. This suggests that maintaining a competitive performance for argument mining task is critical for the multi-task training.

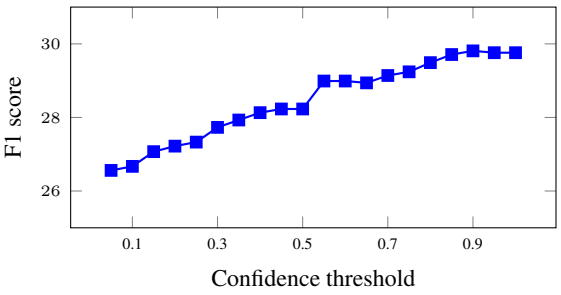

Figure 4: Effect of using different values of the confidence threshold during evaluation.

Comparison of Methods to Obtain Pair Representation. In Section 5.3, we sum up the S-LSTM outputs of each review sentence and rebuttal sentence as the pair representation. We also evaluate the performance when we use concatenation operation to obtain pair representation. It is clear to see from Table 11 that the sum operation significantly outperforms the concatenation operation for the argument pair extraction task as well as two individual subtasks. This suggests that using sum operation here is beneficial to learn better sentence pair representation for argument pair extraction.

B Details on Determining the Argument Pairing Relation

During the evaluation of the argument pair extraction performance, we determine the argument pairing relation on span level with the sentence pairing results. Specifically, we perform binary classification on all sentence pairs enumerated from a candidate argument pair. If more than $p\%$ of sentence pairs are predicted as 1, we say these two arguments form a pair. Here, $p\%$ is a confidence threshold in the range of 0.05 to 1. In the main experiments, $p\%$ is set as 0.9. Take the argument pair $\{$REVIEW-2, REPLY-2$\}$ in Table 1 as an example, REVIEW-2 has 2 sentences, and REPLY-2 has 4 sentences. Thus, we have 8 review-rebuttal sentence pairs. If more than 7.2 (i.e., 0.9×8) sentences are predicted as 1, $\{$REVIEW-2, REPLY-2$\}$ would be identified as an argument pair.

Effect of Confidence Threshold for Argument Pair Extraction. We compare the performance

Models	Argument Mining			Sentence Pairing			Argument Pair Extraction		
	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
MT-LSTM-CRF	61.02	54.72	57.70	72.46	**62.09**	**66.87**	22.09	18.95	20.40
MT-H-LSTM-CRF(L_T=1,L_S=1)	**70.04**	**69.33**	**69.68**	**73.54**	57.96	64.83	28.41	**27.50**	**27.95**
MT-H-LSTM-CRF(L_T=2,L_S=1)	69.46	67.54	68.49	72.17	53.60	61.52	**28.62**	27.25	27.92

Table 7: Performance on RR-passage to compare across different numbers of T-LSTM layers.

Models	Argument Mining			Sentence Pairing			Argument Pair Extraction		
	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
MT-H-LSTM-CRF(L_T=1,L_S=1)	70.04	69.33	69.68	73.54	57.96	**64.83**	28.41	27.50	27.95
MT-H-LSTM-CRF(L_T=1,L_S=2)	**71.85**	**71.01**	**71.43**	72.64	**58.05**	64.53	**30.08**	**29.55**	**29.81**
MT-H-LSTM-CRF(L_T=1,L_S=3)	71.03	68.55	69.77	**74.84**	51.46	60.99	29.37	27.90	28.62

Table 8: Performance on RR-passage to compare across different numbers of S-LSTM layers.

# Linear Layers	Argument Mining			Sentence Pairing			Argument Pair Extraction		
	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
1	**72.03**	68.79	**70.37**	58.24	45.08	50.82	23.25	22.15	22.69
2	69.57	69.31	69.44	71.70	**62.42**	**66.74**	27.37	26.70	27.03
3	70.04	**69.33**	69.68	**73.54**	57.96	64.83	**28.41**	**27.50**	**27.95**
4	69.75	68.71	69.23	69.50	56.52	62.34	26.46	25.55	26.00

Table 9: Performance of **MT-H-LSTM-CRF**(L_T**=1,**L_S**=1**) when using different numbers of linear layers for sentence pairing task.

$\mathcal{L}_1$ Weight	$\mathcal{L}_2$ Weight	Argument Mining			Sentence Pairing			Argument Pair Extraction		
		Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
0.2	0.8	65.82	63.23	64.50	72.78	58.55	64.90	26.52	24.85	25.66
0.3	0.7	69.74	67.62	68.67	73.16	51.31	60.32	28.49	26.85	27.64
0.4	0.6	70.02	69.28	69.65	72.54	51.26	60.07	28.73	28.05	28.38
0.5	0.5	71.85	71.01	71.43	72.64	58.05	64.53	30.08	**29.55**	**29.81**
0.6	0.4	71.90	70.23	71.06	72.59	**59.58**	**65.45**	29.99	28.70	29.33
0.7	0.3	69.50	68.14	68.81	71.09	54.12	61.46	28.04	26.75	27.38
0.8	0.2	**72.17**	**71.29**	**71.72**	**77.55**	48.99	60.05	**30.39**	29.20	29.78
0.9	0.1	66.79	68.44	67.60	70.57	51.07	59.26	25.77	26.30	26.03

Table 10: Performance of **MT-H-LSTM-CRF** when applying different weight ratios between two losses.

Pair Representation	Argument Mining			Sentence Pairing			Argument Pair Extraction		
	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
concatenation	69.68	69.03	69.35	70.86	52.46	60.29	29.76	28.90	29.33
sum	**71.85**	**71.01**	**71.43**	**72.64**	**58.05**	**64.53**	**30.08**	**29.55**	**29.81**

Table 11: Performance of **MT-H-LSTM-CRF** when using different operations to obtain pair representations.

of **MT-H-LSTM-CRF** when using different threshold values. Intuitively, a larger confidence threshold means a stricter criterion for sentence-level alignment for argument pairs. In other words, the confidence of the argument pair prediction is controlled by $p\%$. According to Figure 4, the argument pair extraction performance improves as the threshold value increases. The F1 score reaches the peak when the threshold value is set as 0.9, and drops slightly when the value is closer to 1, as it has an extremely strict requirement for the model to select the correct argument pairs. This suggests that the model is trained to select more accurate argument pairs when the threshold value is larger, while it tends to select more confusing error pair options when the threshold value is smaller.