Argument Pair Extraction via Attention-guided Multi-Layer Multi-Cross Encoding

Liying Cheng[*1,2] **Tianyu Wu**[1] **Lidong Bing**[2] **Luo Si**[2]
[1]Singapore University of Technology and Design [2]DAMO Academy, Alibaba Group
{liying.cheng, l.bing, luo.si}@alibaba-inc.com
tianyu_wu@alumni.sutd.edu.sg

Abstract

Argument pair extraction (APE) is a research task for extracting arguments from two passages and identifying potential argument pairs. Prior research work treats this task as a sequence labeling problem and a binary classification problem on two passages that are directly concatenated together, which has a limitation of not fully utilizing the unique characteristics and inherent relations of two different passages. This paper proposes a novel attention-guided multi-layer multi-cross encoding scheme to address the challenges. The new model processes two passages with two individual sequence encoders and updates their representations using each other's representations through attention. In addition, the pair prediction part is formulated as a table-filling problem by updating the representations of two sequences' Cartesian product. Furthermore, an auxiliary attention loss is introduced to guide each argument to align to its paired argument. An extensive set of experiments show that the new model significantly improves the APE performance over several alternatives [1].

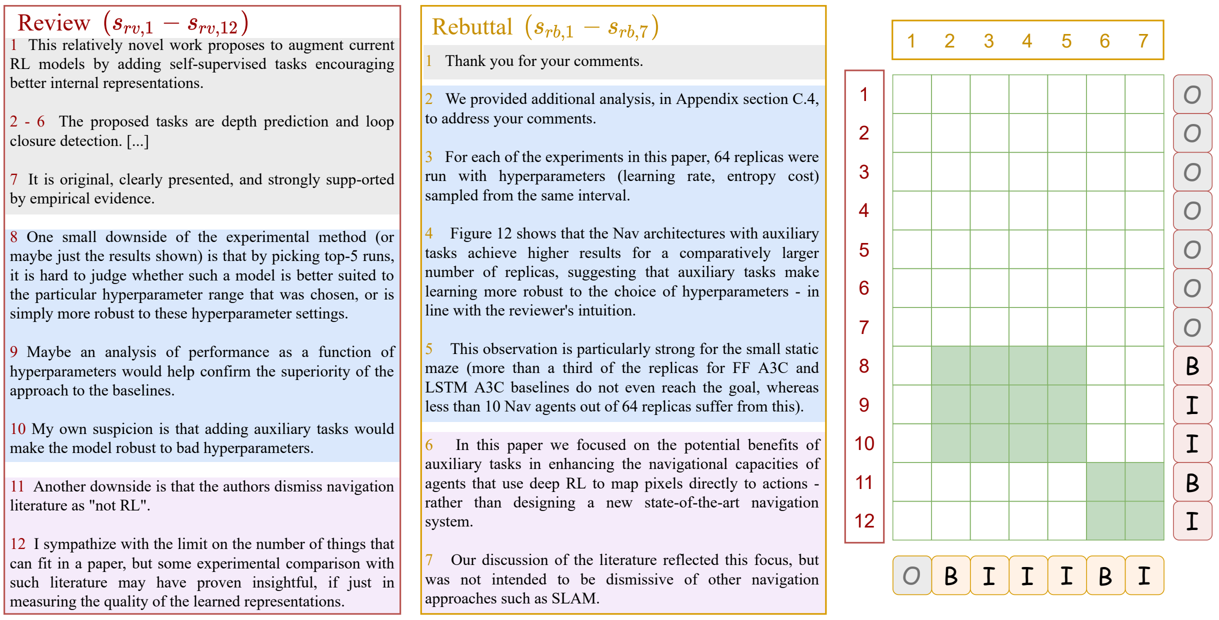
1 Introduction

Mining argumentation structures within a corpus is a crucial task in argument mining research field (Palau and Moens, 2009). There are usually two main components in learning natural language argument structures: (1) detecting argumentative units, (2) predicting relations between the identified arguments. It has been widely studied by natural language processing (NLP) researchers (Cabrio and Villata, 2018) and applied to domains such as: web debating platforms (Boltužić and Šnajder, 2015; Swanson et al., 2015; Chakrabarty et al., 2019), persuasive essays (Stab and Gurevych, 2014; Persing and Ng, 2016), social media (Abbott et al., 2016), etc. Unlike traditional argument extraction tasks that are mainly from monologues, Cheng et al. (2020) propose a new task - argument pair extraction (APE) from two passages in a new domain, namely peer review process, focusing on exploiting the interactions between reviewer comments and author rebuttals. As shown in Figure 1, APE task aims to extract the argument pairs from two passages. Specific suggestions, questions or challenges in reviews are considered as review arguments. Response sentences that answer or explain the specific review argument are its paired rebuttal arguments. For example in the pink area, the reviewer points out the lack of literature review in submission (i.e., review sentences 11-12). As a response, the authors argue that they select the literature based on the special focus of their work (i.e., rebuttal sentence 6-7).

Similar to the two components in the traditional argumentation structure mining, the APE task can be divided into two subtasks: (1) extracting the review and rebuttal arguments from two passages, (2) predicting if an extracted review argument and a rebuttal argument form an argument pair. The first subtask can be cast as a sequence labeling problem and the second one can be cast as a binary classification problem. One straightforward approach is to couple the two subtasks in a pipeline. However, such a pipeline approach learns two subtasks independently without sharing ample information. To address this limitation, the pioneering work (Cheng et al., 2020) employs a multi-task learning framework to train two subtasks simultaneously.

However, there are several shortcomings in the multi-task model. First, the review passage and its rebuttal passage are concatenated as a single passage to perform the argument extraction subtask with sequence labeling. It is obvious to see from

[*]Liying Cheng is under the Joint Ph.D. Program between Alibaba and Singapore University of Technology and Design.
[1]Our code and data are available at https://github.com/TianyuTerry/MLMC.


Review $(s_{rv,1} - s_{rv,12})$

1 This relatively novel work proposes to augment current RL models by adding self-supervised tasks encouraging better internal representations.

2 - 6 The proposed tasks are depth prediction and loop closure detection. [...]

7 It is original, clearly presented, and strongly supp-orted by empirical evidence.

8 One small downside of the experimental method (or maybe just the results shown) is that by picking top-5 runs, it is hard to judge whether such a model is better suited to the particular hyperparameter range that was chosen, or is simply more robust to these hyperparameter settings.

9 Maybe an analysis of performance as a function of hyperparameters would help confirm the superiority of the approach to the baselines.

10 My own suspicion is that adding auxiliary tasks would make the model robust to bad hyperparameters.

11 Another downside is that the authors dismiss navigation literature as "not RL".

12 I sympathize with the limit on the number of things that can fit in a paper, but some experimental comparison with such literature may have proven insightful, if just in measuring the quality of the learned representations.

Rebuttal $(s_{rb,1} - s_{rb,7})$

1 Thank you for your comments.

2 We provided additional analysis, in Appendix section C.4, to address your comments.

3 For each of the experiments in this paper, 64 replicas were run with hyperparameters (learning rate, entropy cost) sampled from the same interval.

4 Figure 12 shows that the Nav architectures with auxiliary tasks achieve higher results for a comparatively larger number of replicas, suggesting that auxiliary tasks make learning more robust to the choice of hyperparameters - in line with the reviewer's intuition.

5 This observation is particularly strong for the small static maze (more than a third of the replicas for FF A3C and LSTM A3C baselines do not even reach the goal, whereas less than 10 Nav agents out of 64 replicas suffer from this).

6 In this paper we focused on the potential benefits of auxiliary tasks in enhancing the navigational capacities of agents that use deep RL to map pixels directly to actions - rather than designing a new state-of-the-art navigation system.

7 Our discussion of the literature reflected this focus, but was not intended to be dismissive of other navigation approaches such as SLAM.

	1	2	3	4	5	6	7	
1								O
2								O
3								O
4								O
5								O
6								O
7								O
8		🟩	🟩	🟩	🟩			B
9		🟩	🟩	🟩	🟩			I
10		🟩	🟩	🟩	🟩			I
11						🟩	🟩	B
12						🟩	🟩	I
	O	B	I	I	I	B	I	

Figure 1: An example of APE task. The review and rebuttal passage pair is shown on the left. The grey area refers to non-arguments, while the blue and pink areas refer to two paired arguments. The table representing the pairing relation is shown on the right (filled entries: paired; unfilled entries: unpaired). Review and rebuttal sentence indices are on the left and the top of the table. Review and rebuttal sequence labels for argument extraction are on the right and the bottom of the table.

Figure 1 that the review and rebuttal passages have their own styles in terms of structure and wording. Hence, it is not suitable to concatenate them as one long sequence, which is against the fact that they are two unique sequences in essence and hinders the model from well-utilizing their different characteristics. To overcome this limitation, we treat review and rebuttal passages as two individual sequences and design two sequence encoders for them respectively. In each sequence encoder, the sequence representations will be updated by the other's representations through mutual attention. It allows us to better distinguish two passages, and meanwhile, to conveniently exchange information between them through the attention mechanism.

Second, the subtask coordination capability of their multi-task framework is weak as two subtasks only coordinate with each other via the shared feature encoders, i.e., the sentence encoder for the sequence of word tokens and the passage encoder for the concatenation of sentences. Thus, the shared information between two subtasks is only learned implicitly. To overcome this limitation, we propose an attention-guided multi-layer multi-cross (MLMC) encoding mechanism. Inspired by the table-filling approach (Miwa and Sasaki, 2014), we form a table that represents features for the Cartesian Product of review and rebuttal sequences by utilizing both of their embeddings, as shown in the right portion of Figure 1. The table representations will be updated with the incorporation of the two sequence representations, and in return, it will also help to update the mutual attention mentioned above. It is named as multi-cross encoder because these three encoding components (i.e., one table and two sequences) interact with each other explicitly and extensively. By stacking multiple encoder layers, the two subtasks can further benefit each other. In addition, we also design an auxiliary attention loss to guide each argument to refer to its paired arguments. This additional loss not only enhances the model performance, but also significantly improves the attention interpretability.

To summarize, the contributions of this paper are three-fold. Firstly, we apply the table-filling approach to model the sentence-level correlation between two passages with multiple sentences for the first time. Secondly, on the model side, we propose an MLMC encoder to explicitly learn the useful shared information in the two passages. Furthermore, we introduce an auxiliary attention loss, which is able to further improve the efficacy of the mutual attentions. Thirdly, we evaluate our model on the benchmark dataset (Cheng et al., 2020), and the results show that our model achieves a new state-of-the-art performance on the APE task.

2 Related Work

Argument mining has wide applications in educational domain, including persuasive essays (Stab and Gurevych, 2017; Eger et al., 2017), scientific articles (Teufel et al., 2009; Guo et al., 2011), writing assistance (Zhang and Litman, 2016), essay scoring (Persing and Ng, 2015; Somasundaran et al., 2016), peer reviews (Hua et al., 2019), etc. Unlike previous works, Cheng et al. (2020) introduce a new task named APE in the domain of peer review and rebuttal, which intends to extract the argument pairs from two passages simultaneously.

Table-filling approaches (Miwa and Sasaki, 2014; Gupta et al., 2016; Zhang et al., 2017) have been proposed to work towards the joint task of name entity recognition (NER) and relation extraction (RE). In their work, the diagonal entries of the table show the words' entity types and the off-diagonal entries show the relation types with other words. More recently, there are more research works to propose various table-filling models on different tasks. Wang and Lu (2020) propose to learn two separate encoders (a table encoder and a sequence encoder) by interacting with each other for joint NER and RE task. Wu et al. (2020) propose a grid tagging scheme to address the aspect-oriented fine-grained opinion extraction task. Compared to our model, one major difference is the table shape. In their tables, the row and column represent the same sequence, and thus in square shape. In our model, the table is in a rectangle shape where the row and column represent two different sequences with different lengths. Another clear difference is that each entry in their table is for word-pair relation, whereas each entry in our table captures sentence-pair relation. As we can see from Figure 1, the review/rebuttal sequence consists of a list of sentences. Thus, it requires extra effort to learn comprehensive sentence representations.

3 Task Formulation

In this paper, we tackle the APE task, which aims to study the internal structure and relations between two passages, e.g., review and rebuttal passages. For example, as shown in Figure 1, given a pair of review passage $\mathbf{s_{rv}} = [s_{rv,1}, \cdots, s_{rv,12}]$ (in the red box) and rebuttal passage $\mathbf{s_{rb}} = [s_{rb,1}, \cdots, s_{rb,7}]$ (in the orange box), we intend to automatically extract all argument pairs between them. First, for the argument mining subtask, we cast it as a sentence-level sequence labeling problem following the work (Cheng et al., 2020) using the standard BIO scheme (Ramshaw, 1995; Ratinov and Roth, 2009). This subtask segments the argumentative units (highlighted in blue/pink) from non-argumentative units (highlighted in grey) for each passage. The label sequences for the review passage and the rebuttal passage are shown in the right portion of Figure 1. Second, the sentence pairing subtask predicts whether the two sentences belong to one argument pair. Here, we formulate it as a table-filling problem following the work (Miwa and Sasaki, 2014). Take the $8th$ review sentence $s_{rv,8}$ in the first review argument as an example, the rebuttal argument sentences $\{s_{rb,2}, s_{rb,3}, s_{rb,4}, s_{rb,5}\}$ forming sentence pairs with it are filled with green, as shown in the table. With the collaboration of these two subtasks, we can perform the overall argument pair extraction task. In this case, two argument pairs (highlighted in blue/pink from two passages) are extracted, which correspond to the two green rectangles shown in the table.

4 Model

Figure 2 shows our proposed attention-guided multi-layer multi-cross (MLMC) encoding based model. The model mainly consists of three parts: a sentence embedder, an n-layer multi-cross encoder, and a predictor. The review sentences and rebuttal sentences first go through the sentence embedder separately to obtain their sentence embeddings respectively. We then utilize the representations from review and rebuttal sequences to form a table as shown earlier in Figure 1. Next, the representations of the table and two sequences are updated through n multi-cross encoder layers. Finally, the model predicts the review and rebuttal arguments through a conditional random field (CRF) (Lafferty et al., 2001) layer based on two sequence representations, and extracts the pairing information through a multi-layer perceptron (MLP) based on the table representations.

4.1 Sentence Embedder

The bottom left part of Figure 2 shows our sentence embedder, the input of which is a review sentence or a rebuttal sentence with l tokens $s = [t_0, t_1, \cdots, t_{l-1}]$. We obtain the pre-trained BERT (Devlin et al., 2019) token embeddings $[x_0, x_1, \cdots, x_{l-1}]$ for all word tokens in the sentence, after which all token embeddings are fed into a bidirectional long short-term memory (biLSTM)

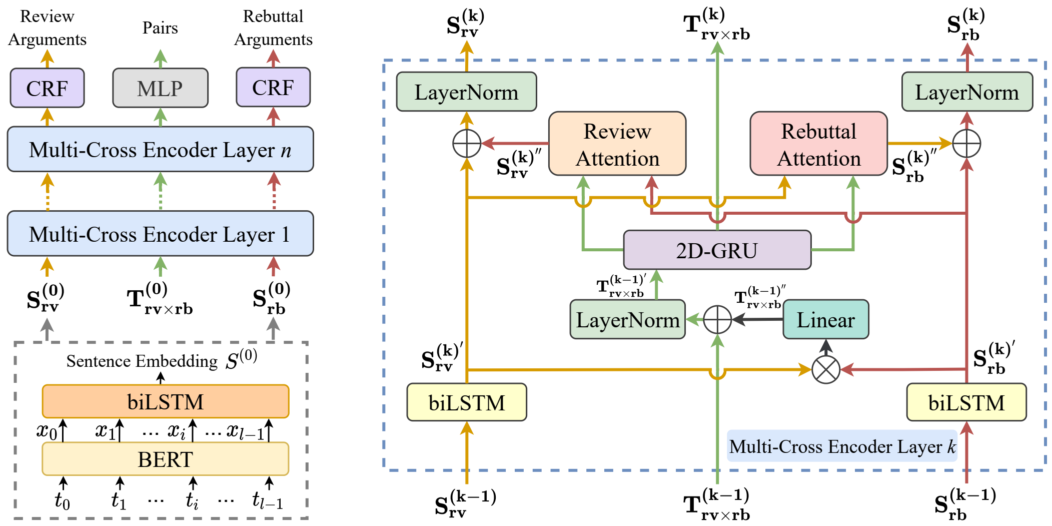


Figure 2: Overview of our model architecture with n multi-cross encoder layers (shown on the left). The sentence embedder (in the grey dotted box at the bottom left) shows the process of obtaining initial review and rebuttal sentence embeddings from pre-trained BERT token embeddings with biLSTM. The kth multi-cross encoder layer (in the blue dotted box on the right) shows the process of getting the sentence representations of review and rebuttal and the pair representations for the next layer.

(Hochreiter and Schmidhuber, 1997) layer. The last hidden states from both directions are concatenated as the sentence embedding $S^{(0)}$. A more common practice is to use the [CLS] token embedding to represent the sentence embedding. However, given the high density of scientific terms and the correspondence between review and rebuttal, token-level information is naturally crucial for the task. The same conclusion is drawn by the experimental results in the previous work (Cheng et al., 2020).

4.2 Multi-Cross Encoder

The entire multi-cross encoder consists of n layers. The details of each multi-cross encoder layer are shown in the blue dotted box on the right of Figure 2. The input of the layer includes table representations and two sequence representations, i.e., review and rebuttal sequence representations. In each layer, table features are updated by sequence features and vice versa.

Sequence Encoder Phase I To well-utilize different characteristics of review and rebuttal, we regard them as two individual sequences. Two sequence embeddings $\mathbf{S}_{\mathbf{rv}}^{(\mathbf{k-1})}$ and $\mathbf{S}_{\mathbf{rb}}^{(\mathbf{k-1})}$ of length I and J respectively (i.e., the output from the previous layer) are passed through the same biLSTM layer colored light yellow in Figure 2. Take review sequence as an example, the review hidden states at position i are updated as follows:

$$S_{rv,i}^{(k)'[1]} = \mathrm{LSTM}_{\mathrm{forward}}(S_{rv,i}^{(k-1)}, S_{rv,i-1}^{(k)'[1]}),$$

$$S_{rv,i}^{(k)'[2]} = \mathrm{LSTM}_{\mathrm{backward}}(S_{rv,i}^{(k-1)}, S_{rv,i+1}^{(k)'[2]}),$$

$$S_{rv,i}^{(k)'} = [S_{rv,i}^{(k)'[1]}, S_{rv,i}^{(k)'[2]}].$$

The rebuttal hidden states $\mathbf{S}_{\mathbf{rb}}^{(\mathbf{k})'}$ in layer k is obtained from the same biLSTM in the same manner.

Table Encoder To capture the pairing information explicitly, we adopt the table-filling approach. At layer k, we update the table $\mathbf{T}_{\mathbf{rv\times rb}}^{(\mathbf{k-1})}$ through the table encoder. The table input $\mathbf{T}_{\mathbf{rv\times rb}}^{(\mathbf{0})}$ before the first encoder layer are set as $\mathbf{0}$. At each layer k, in order to incorporate the information extracted in $\mathbf{S}_{\mathbf{rv}}^{(\mathbf{k})'}$ and $\mathbf{S}_{\mathbf{rb}}^{(\mathbf{k})'}$, we form another table $\mathbf{T}_{\mathbf{rv\times rb}}^{(\mathbf{k-1})''}$ with them through concatenation and linear projection as follows:

$$\mathbf{T}_{\mathbf{rv\times rb}}^{(\mathbf{k-1})''} = \mathrm{Linear}(\mathbf{S}_{\mathbf{rv}}^{(\mathbf{k})'} \otimes \mathbf{S}_{\mathbf{rb}}^{(\mathbf{k})'}).$$

The table features from previous layer $\mathbf{T}_{\mathbf{rv\times rb}}^{(\mathbf{k-1})}$ are then updated by $\mathbf{T}_{\mathbf{rv\times rb}}^{(\mathbf{k-1})''}$ with layer normalization:

$$\mathbf{T}_{\mathbf{rv\times rb}}^{(\mathbf{k-1})'} = \mathrm{LayerNorm}(\mathbf{T}_{\mathbf{rv\times rb}}^{(\mathbf{k-1})} \oplus \mathbf{T}_{\mathbf{rv\times rb}}^{(\mathbf{k-1})''}).$$

The entry $T_{i,j}^{(k-1)'}$ at row i and column j represents specific features between review sentence at position i and rebuttal sentence at position j. The table hidden states $T_{i,j}^{(k)}$ are updated through 2D-GRU:

$$T_{i,j}^{(k)[1]} = \mathrm{GRU}_{\mathrm{forward}}(T_{i-1,j}^{(k)[1]}, T_{i,j-1}^{(k)[1]}, T_{i,j}^{(k-1)'}),$$

$$T_{i,j}^{(k)[2]} = \mathrm{GRU}_{\mathrm{backward}}(T_{i+1,j}^{(k)[2]}, T_{i,j+1}^{(k)[2]}, T_{i,j}^{(k-1)'}),$$

$$T_{i,j}^{(k)} = [T_{i,j}^{(k)[1]}, T_{i,j}^{(k)[2]}].$$

The 2D-GRU settings are similar to the previous work (Wang and Lu, 2020) except that the table to be processed is not necessarily a square ($I \neq J$ in general). Therefore, the 2D-GRU implemented here is more general. The previous hidden states for table boundaries ($T_{0,j}^{(k)[1]}, T_{i,0}^{(k)[1]}, T_{I+1,j}^{(k)[2]}, T_{i,J+1}^{(k)[2]}$) are set as $\mathbf{0}$. The outputs $\mathbf{T}_{\mathbf{rv}\times\mathbf{rp}}^{(\mathbf{k})}$ of layer k are further exploited by the mutual attention mechanism explained below to update review and rebuttal sequence embeddings.

Mutual Attention The mutual attention mechanism (shown as review attention and rebuttal attention modules in Figure 2) links review embedding, rebuttal embedding and table embedding together, through which review embedding and rebuttal embedding update each other with the help of table features. The attention weights $\alpha_{i,j}^{(k)}$ and $\beta_{i,j}^{(k)}$ at position (i, j) in layer k are updated as follows:

$$\alpha_{i,j}^{(k)} = \tanh(v_\alpha^T \cdot T_{i,j}^{(k)}), \quad \beta_{i,j}^{(k)} = \tanh(v_\beta^T \cdot T_{i,j}^{(k)}),$$

where v_α and v_β are learnable vectors. We further normalize the attention weights:

$$a_{i,j}^{(k)} = \frac{\exp(\alpha_{i,j}^{(k)})}{\sum_{j'=1}^{J} \exp(\alpha_{i,j'}^{(k)})}, \quad b_{i,j}^{(k)} = \frac{\exp(\beta_{i,j}^{(k)})}{\sum_{i'=1}^{I} \exp(\beta_{i',j}^{(k)})}.$$

Here, $a_{i,j}^{(k)}$ and $b_{i,j}^{(k)}$ are the normalized attention weights ranging from 0 to 1. We then get the weighted average of sentence representations $S_{rv,i}^{(k)''}$ and $S_{rb,j}^{(k)''}$ from $\mathbf{S}_{\mathbf{rb}}^{(\mathbf{k})'}$ and $\mathbf{S}_{\mathbf{rv}}^{(\mathbf{k})'}$ respectively.

$$S_{rv,i}^{(k)''} = \sum_{j=1}^{J} a_{i,j}^{(k)} \cdot S_{rb,j}^{(k)'}, \quad S_{rb,j}^{(k)''} = \sum_{i=1}^{I} b_{i,j}^{(k)} \cdot S_{rv,i}^{(k)'}.$$

Here, $\mathbf{S}_{\mathbf{rv}}^{(\mathbf{k})''}$ and $\mathbf{S}_{\mathbf{rb}}^{(\mathbf{k})''}$ are the updated review embedding and rebuttal embedding. Information in review and rebuttal sequences is exchanged via mutual attention.

Sequence Encoder Phase II The addition and layer normalization used to combine $\mathbf{S}^{(\mathbf{k})'}$ and $\mathbf{S}^{(\mathbf{k})''}$ in the sequence encoder are similar to the one in table encoder. We obtain the review sequence embedding $\mathbf{S}_{\mathbf{rv}}^{(\mathbf{k})}$ and rebuttal sequence embedding $\mathbf{S}_{\mathbf{rb}}^{(\mathbf{k})}$ as the sequence outputs of layer k as follows:

$$\mathbf{S}_{\mathbf{rv}}^{(\mathbf{k})} = \text{LayerNorm}(\mathbf{S}_{\mathbf{rv}}^{(\mathbf{k})'} \oplus \mathbf{S}_{\mathbf{rv}}^{(\mathbf{k})''}),$$

$$\mathbf{S}_{\mathbf{rb}}^{(\mathbf{k})} = \text{LayerNorm}(\mathbf{S}_{\mathbf{rb}}^{(\mathbf{k})'} \oplus \mathbf{S}_{\mathbf{rb}}^{(\mathbf{k})''}).$$

Stacking Multi-Cross Encoder Layers The updating process described above continues as layer grows from 1 to n. The table feature is updated by both review and rebuttal sequences, and each sequence updates the other via the table later on. There are also residual connections between adjacent layers which accept the previous layer's output as the current layer's input and include it as part of the new embedding, making the system more robust. All three features (i.e., review sequence, rebuttal sequence, table) are intertwined with each other and information flows across different components of the encoder. This is also the reason why the encoder is described as MLMC.

4.3 Argument Pair Predictor

After the final multi-cross encoder layer, sequence features are used for argument mining and table features are used for pair prediction.

Argument Predictor We adopt CRF to predict argument sequence labels. The sequence labeling loss $\mathcal{L}_{seq}$ for both review sequence $\mathbf{s}_{\mathbf{rv}}$ and rebuttal sequence $\mathbf{s}_{\mathbf{rb}}$ in each instance is defined as:

$$\mathcal{L}_{seq} = -\big(\log p(\mathbf{y}_{\mathbf{rv}}|\mathbf{s}_{\mathbf{rv}}) + \log p(\mathbf{y}_{\mathbf{rb}}|\mathbf{s}_{\mathbf{rb}})\big),$$

where $\mathbf{y}_{\mathbf{rv}}$ and $\mathbf{y}_{\mathbf{rb}}$ are the review and rebuttal sequence labels [2].

During inference, the predicted sequence label is the one with the highest conditional probability given the original sequence:

$$\mathbf{y}_{\mathbf{rv}}^{*} = \arg\max_{\mathbf{y}} p(\mathbf{y}|\mathbf{s}_{\mathbf{rv}}), \quad \mathbf{y}_{\mathbf{rb}}^{*} = \arg\max_{\mathbf{y}} p(\mathbf{y}|\mathbf{s}_{\mathbf{rb}}).$$

Pair Predictor We use MLP to predict sentence pairs [3]. The pairing loss $\mathcal{L}_{pair}$ for each instance is:

$$\mathcal{L}_{pair} = -\sum_{i,j} \Big(y_{i,j}^{pair} \log p(y_{i,j}^{pair} = 1|\mathbf{s}_{\mathbf{rv}}, \mathbf{s}_{\mathbf{rb}}) + (1 - y_{i,j}^{pair}) \log p(y_{i,j}^{pair} = 0|\mathbf{s}_{\mathbf{rv}}, \mathbf{s}_{\mathbf{rb}})\Big),$$

where $y_{i,j}^{pair}$ is 1 when $s_{rv,i}$ and $s_{rb,j}$ are paired, and is 0 otherwise [4].

Following (Cheng et al., 2020), during evaluation, a pair of candidate spans ($[s_{rv,i_1}, \cdots, s_{rv,i_2}]$ and $[s_{rb,j_1}, \cdots, s_{rb,j_2}]$) form a pair if they satisfy the following criterion:

$$\frac{\sum_{i=i_1}^{i_2} \sum_{j=j_1}^{j_2} \mathbb{1}_{\{p(y_{i,j}^{pair}=1)>0.5\}}}{(i_2-i_1+1)\times(j_2-j_1+1)} \times 100\% \geqslant 50\%$$

Attention Loss Attention loss is a loss term specifically designed for the task. It aims to increase the effectiveness of review attention and rebuttal attention discussed above. Even without

[2]We provide the detailed steps of deriving the loss $\mathcal{L}_{seq}$ in Appendix A.1.

[3]MLP is chosen because more complex structures like convolutional neural networks (CNN) demonstrate no superiority. The comparison results are attached in Appendix B.3.

[4]We provide the detailed steps of deriving the pairing loss $\mathcal{L}_{pair}$ in Appendix A.2.

this auxiliary loss term, sentences in review are supposed to attend to relevant sentences in rebuttal and vice versa. The auxiliary loss is thus aimed at augmenting the effect of mutual reference explicitly by guiding the paired arguments to refer to each other. Intuitively, under the settings of argument mining and pairing, it is natural that review arguments refer to the paired rebuttal arguments to update their embedding and vice versa during mutual attention. Hence, we introduce an auxiliary loss term to increase the attention weights computed for paired arguments and decrease the attention weights otherwise for both review and rebuttal attentions in all layers. For each instance, $\mathcal{L}_{attn}$ is defined as:

$$\mathcal{L}_{attn} = \sum_{i,j}(1 - 2y_{i,j}^{pair}) \cdot \Big(\sum_{k=1}^{n} \gamma^{n-k} \cdot (a_{i,j}^{(k)} + b_{i,j}^{(k)})\Big),$$

where γ is the decaying parameter used to compute exponential moving average for the sum of attention. Larger weights are assigned to layers closer to the final predictor as they are more related to the prediction in the end. The attention loss is defined in the form of summation across all layers to increase the accuracy and interpretability of both review and rebuttal attentions in all layers. If the tendency to attend to the paired argument is augmented, the benefits of attention mechanism can be further exploited (e.g., learning better sentence representations, increasing pair prediction accuracy).

The overall loss $\mathcal{L}$ is then defined by summing up three losses together:

$$\mathcal{L} = \mathcal{L}_{seq} + \lambda_1 \cdot \mathcal{L}_{pair} + \lambda_2 \cdot \mathcal{L}_{attn},$$

where λ_1 and λ_2 are tuned hyperparameters.

5 Experiments

5.1 Data

We conduct experiments on the benchmark dataset, i.e., RR dataset (Cheng et al., 2020) to evaluate the effectiveness of our proposed model. RR dataset includes 4,764 pairs of peer reviews and author rebuttals collected from ICLR 2013 to ICLR 2020. There are two dataset versions provided: RR-Submission-v1 and RR-Passage-v1. In RR-Submission-v1, multiple review-rebuttal passage pairs of the same paper submission are in the same set of train, dev or test; while in RR-Passage-v1, different review-rebuttal passage pairs of the same submission could be put into different sets. We further modify the RR-Submission-v1 dataset by fixing some minor bugs in the labels, and name it RR-Submission-v2. The data are split into train, dev and test sets by a ratio of 8:1:1 for all three dataset versions.

5.2 Baselines

We compare our model with two baselines:

- The pipeline approach is used as a baseline model in the previous work (Cheng et al., 2020). It independently trains two subtasks and then pipes them together to extract argument pairs.
- The multi-task learning model proposed by (Cheng et al., 2020) trains two subtasks simultaneously via the shared feature encoders.

5.3 Experimental Settings

We implement our attention-guided MLMC encoding based model in Pytorch. The dimension of pre-trained BERT sentence embeddings is 768 by default. Maximum number of BERT tokens for each sentence is set as 200. MLP layer is composed of 3 linear functions and 2 ReLU functions. We use Adam (Kingma and Ba, 2014) with an initial learning rate of 0.0002, and update parameters with a batch size of 1 and dropout rate of 0.5. We train our model for 25 epochs at most. We select the best model parameters based on the best overall F_1 score on the development set and apply it to the test set for evaluation. All models are run with V100 GPU. Note that in this paper, the parameters are mainly tuned based on RR-Submission-v1 [5]. Following the previous work (Cheng et al., 2020), we report the precision (Prec.), recall (Rec.) and F_1 scores for the performance on both subtasks as well as the overall extraction performance.

5.4 Main Results on RR Dataset

Table 1 shows the performance comparison between our proposed models and the pervious work on RR-Submission-v1 and RR-Passage-v1 datasets [6]. Besides the two baseline models mentioned before, we implement a bi-cross encoding scheme (Bi-Cross) for comparisons as well. The key difference between the bi-cross encoder and the multi-cross encoder is that in the bi-cross encoder,

[5]More details about hyperparameter settings (e.g. weight for pair loss λ_1, weight for attention loss λ_2, decaying parameter γ of exponential moving average) and experimental results (e.g. running time, number of parameters, performance on the development set) could be found in Appendix B.

[6]The previous work adopts negative sampling technique for sentence pairing subtask and evaluates the performance on the partial test set. In this work, we re-evaluate the previous work's sentence pairing subtask on the whole test dataset for a fair comparison. Those results are marked with * in Table 1.

Data	Models	Argument Mining			Sentence Pairing			APE		
		Prec.	Rec.	F_1	Prec.	Rec.	F_1	Prec.	Rec.	F_1
RR-Submission-v1	Pipeline (Cheng et al., 2020)	67.63	68.51	68.06	50.05*	47.15*	48.56*	19.86	19.94	19.90
	Multi-task (Cheng et al., 2020)	70.09	70.14	70.12	53.44*	42.71*	47.48*	26.69	26.24	26.46
	Bi-Cross (n=1)	68.13	69.69	68.90	57.14	42.28	48.60	35.56	22.95	27.90
	Bi-Cross (n=2)	70.38	68.12	69.23	61.93	43.56	51.15	38.36	23.34	29.02
	Bi-Cross (n=3)	68.04	71.65	69.80	62.73	42.33	50.55	39.43	24.45	30.18
	Bi-Cross (n=4)	67.45	71.48	69.41	61.19	43.88	51.11	36.59	25.28	29.90
	Bi-Cross (n=5)	68.47	70.30	69.37	57.29	41.30	48.00	36.37	24.06	28.96
	Multi-Cross (n=1)	67.64	69.27	68.45	60.08	44.33	51.01	36.56	23.62	28.70
	Multi-Cross (n=2)	67.23	70.55	68.85	62.57	43.71	51.46	37.50	25.39	30.28
	Multi-Cross (n=3)	69.52	70.03	69.77	60.81	47.14	53.11	**38.70**	**27.93**	**32.44**
	Multi-Cross (n=4)	65.84	70.60	68.14	60.07	40.59	48.45	35.98	26.33	30.41
	Multi-Cross (n=5)	65.52	70.79	68.05	60.69	44.90	51.61	36.32	27.82	31.51
RR-Passage-v1	Pipeline (Cheng et al., 2020)	73.10	67.65	70.27	51.34*	42.08*	46.25*	21.24	19.30	20.23
	Multi-task (Cheng et al., 2020)	71.85	71.01	71.43	54.28*	43.24*	48.13*	30.08	29.55	29.81
	Bi-Cross (n=1)	67.77	70.91	69.31	62.05	41.27	49.57	40.72	25.74	31.54
	Bi-Cross (n=2)	69.44	71.59	70.50	60.20	43.26	50.34	41.48	27.53	33.09
	Bi-Cross (n=3)	66.79	71.97	69.28	61.22	43.02	50.53	39.04	26.89	31.85
	Bi-Cross (n=4)	67.55	71.95	69.70	62.35	39.98	48.72	**42.02**	27.32	33.11
	Bi-Cross (n=5)	66.32	71.35	68.74	62.72	37.70	47.09	40.85	25.26	31.22
	Multi-Cross (n=1)	67.18	72.29	69.64	61.29	45.94	52.52	38.82	29.32	33.40
	Multi-Cross (n=2)	68.28	72.12	70.15	61.54	44.23	51.47	40.13	29.11	33.74
	Multi-Cross (n=3)	66.79	72.17	69.37	62.69	42.33	50.53	40.27	**29.53**	**34.07**
	Multi-Cross (n=4)	65.77	73.54	69.44	62.80	39.98	48.86	38.96	27.32	32.12
	Multi-Cross (n=5)	67.84	71.42	69.58	60.94	41.08	49.08	39.60	26.95	32.07

Table 1: Main results on RR-Submission-v1 and RR-Passage-v1. The results with * are re-evaluated.

Data	Models	Argument Mining			Sentence Pairing			APE		
		Prec.	Rec.	F_1	Prec.	Rec.	F_1	Prec.	Rec.	F_1
RR-Submission-v2	Multi-task (Cheng et al., 2020)	70.74	69.46	70.09	52.05	46.74	49.25	27.24	26.00	26.61
	Bi-Cross (n=1)	68.46	70.47	69.45	66.10	41.25	50.80	37.86	24.78	29.95
	Bi-Cross (n=2)	68.69	72.54	70.56	59.40	47.61	52.86	35.17	27.42	30.81
	Bi-Cross (n=3)	69.04	72.35	70.66	58.87	44.10	50.43	35.68	26.26	30.25
	Bi-Cross (n=4)	68.78	72.54	70.61	61.72	43.87	51.28	35.54	26.68	30.48
	Multi-Cross (n=1)	69.20	72.30	70.72	60.92	43.10	50.48	36.18	26.68	30.71
	Multi-Cross (n=2)	69.13	72.66	70.85	59.94	47.60	53.06	34.97	29.01	31.71
	Multi-Cross (n=3)	69.53	73.27	71.35	60.01	46.82	52.60	37.15	**29.38**	**32.81**
	Multi-Cross (n=4)	69.73	71.36	70.54	62.87	48.15	54.53	36.17	28.53	31.90
	Multi-Cross (n=5)	67.80	71.36	69.54	64.31	43.91	52.19	**39.56**	27.69	32.58

Table 2: Main results on RR-Submission-v2.

the review sentences and rebuttal sentences are concatenated as one sequence, and thus it only has one sequence encoder. In contrast, there are two individual sequence encoders in our multi-cross encoder. With the same number of layers, our multi-cross model outperforms the bi-cross model on both datasets except for RR-Passage-v1 with 4 layers. This is especially conspicuous when the number of layers is 3. The superiority of the multi-cross model demonstrates the importance and robustness of learning review and rebuttal sequences separately. Our model achieves the highest F_1 score when the number of layers increases to 3. Adding more layers hurts the performance, probably because the model overfits with too many layers. Table 2 shows the performance on RR-Submission-v2 [7]. The main conclusion is consistent with the performace on RR-Submission-v1. Both the bi-cross and multi-cross models outperform the multi-task model, and the multi-cross models further outperform the bi-cross models.

Although the baselines achieve slightly better performance on the argument mining subtask than both the bi-cross model and the multi-cross model, they still perform worse than our models on the sentence pairing subtask and the overall APE task. This is plausibly because of two main reasons. First, in the multi-task model, the subtask coordi-

[7] We encourage the researchers to use RR-submission-v2 and compare to its performance in the future.

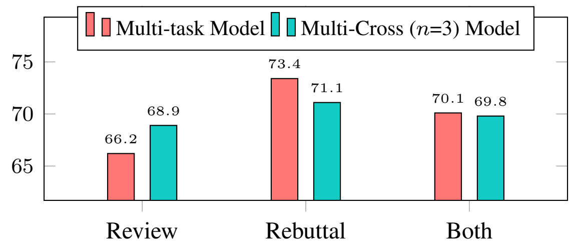


Figure 3: F_1 scores of the argument mining subtask.

Model Settings	APE			
	Prec.	Rec.	F_1	Δ (F_1)
Multi-Cross (n=3)	38.70	27.93	32.44	-
w/o sharing biLSTM	36.95	26.00	30.52	-1.92
w/o sharing CRF	33.80	28.10	30.69	-1.75
w/o sharing both	36.86	27.38	31.42	-1.02
w/o cross update	38.31	25.55	30.66	-1.78
w/o attention loss	37.15	24.56	29.57	-2.87

Table 3: Ablation study of multi-cross (n=3) model on RR-Submission-v1 dataset.

nation capability is weak as the shared information between two subtasks is learned implicitly. However, in our model, the three encoding components are explicitly mingled with each other through the mutual attention mechanism and the table encoder. On one hand, the better sentence pairing subtask performance demonstrates the effectiveness of the table-filling approach. On the other hand, the better overall APE performance demonstrates the strong subtask coordination capability of our model architecture. Second, we further analyze the breakdown performance of the multi-task model and our multi-cross (n=3) model on the argument mining subtask. Figure 3 shows the subtask performance on RR-Submission-v1 dataset for reviews, rebuttals, and both of them. We can observe that the difference of F_1 scores between reviews and rebuttals of our model is smaller than the multi-task model. Despite the slight decrease in the overall argument mining performance, a more balanced argument extraction performance on reviews and rebuttals brings in better overall APE performance, which is because more accurate review argument extraction increases the chance for the extracted rebuttal arguments to be paired correctly.

5.5 Ablation study

We conduct an ablation study of the multi-cross (n=3) model on RR-Submission-v1 dataset from three perspectives, as presented in Table 3. Firstly, we evaluate the effect of sharing the biLSTM layer (the light yellow modules in Figure 2) and the CRF layer. We can notice that the F_1 drops 1.92 without sharing the biLSTM layer, drops 1.75 without sharing the CRF layer, and drops 1.02 when sharing neither. It is interesting to notice that when two sequences use their own biLSTMs and CRF simultaneously (i.e., w/o sharing both), the F_1 drops less compared to the models without sharing only one of them. This suggests that having an individual set of biLSTM and CRF layers for each type of sequence is plausibly a worthwhile setting, but it is not as effective as sharing both. One possible reason is that the advantage brought in by such a tailor-made sequential tagging configuration for each type is overwhelmed by the disadvantage of fewer training instances. Secondly, without cross updates between the review and rebuttal embeddings (the mutual attention modules still exist), the F_1 drops 1.78. This result again demonstrates the effectiveness of explicitly blending two sequence embeddings via the mutual attention mechanism specifically designed for this task. Thirdly, we also investigate the effect of attention loss term by removing it from the overall loss. The performance drops about 2.87 F_1 points. We will elaborate more with the attention visualization below.

5.6 Attention Visualization

To examine the effectiveness of the auxiliary attention loss, we visualize the sum of attention weights of all layers for four test samples, as shown in Figure 4. The sum is computed for visualization because attention weights in all layers are guided by the attention loss. The distribution of attention is significantly improved as the colors for arguments in Column (c) are considerably darker. In Column (b) without the guidance of attention loss, despite some patterns, attention weights are distributed in a quite haphazard manner. Therefore, the interpretability of our model is much better as we can easily understand which part of the discourse each sentence refers to. Specifically, the boundary of most attention blocks in Column (c) matches well with the start and end positions of the ground truth review and rebuttal arguments. The gold and predicted argument spans and argument pairs of these four samples are shown in Appendix C.1, and more discussions are given regarding the reason for some mistakenly predicted boundaries. The effectiveness of the auxiliary attention loss is also quantitatively illustrated by a higher F_1 score after its incorporation (32.44 v.s. 29.57) in Table 3.

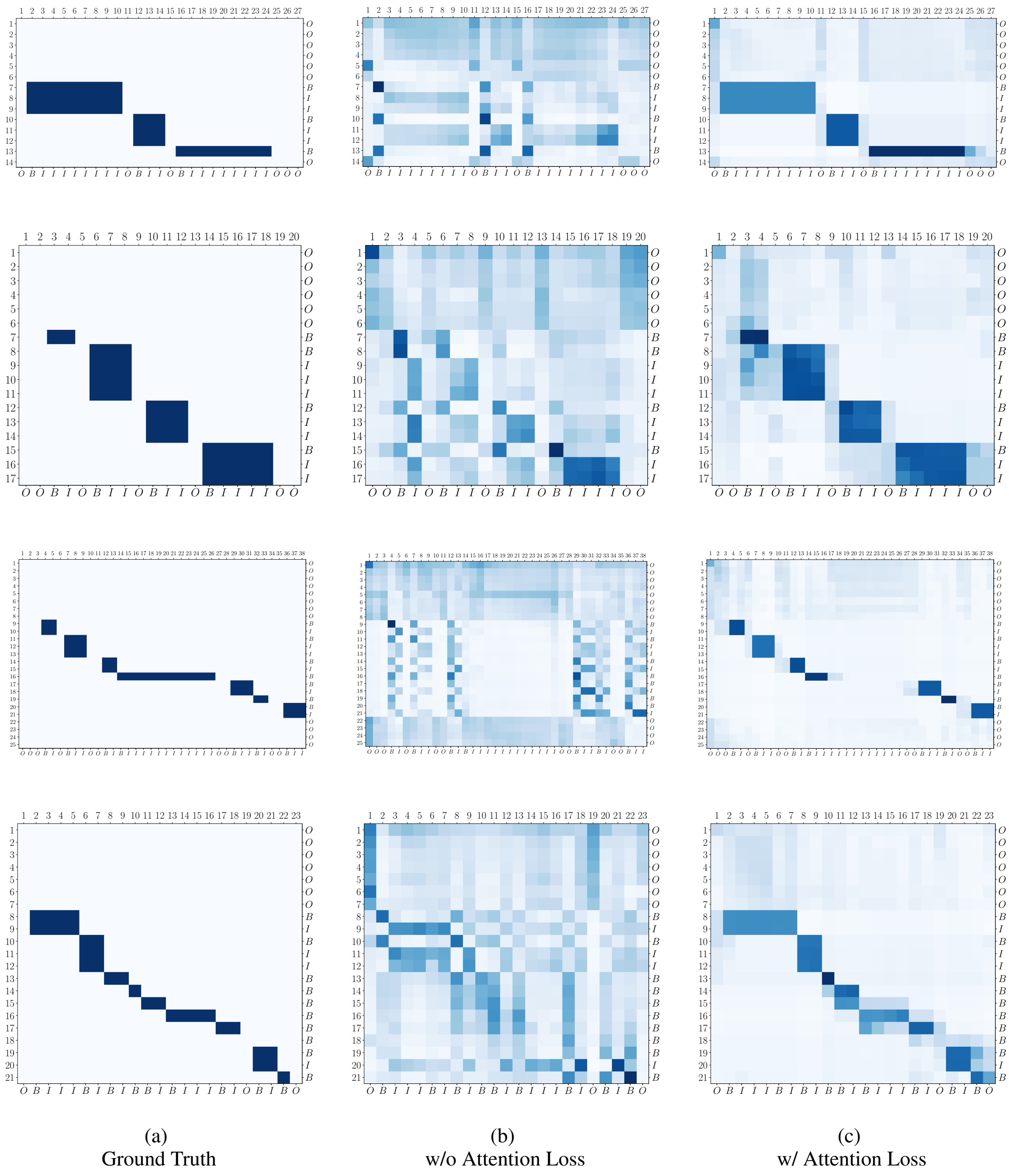


Figure 4: Visualization of attention weights in four test data samples. In each plot, the y-axis corresponds to the review sentences and the x-axis corresponds to the rebuttal sentences. Each pixel shows the attention weight of a review and rebuttal sentence. (a) Ground truth: the gold labels of argument pairs; (b) w/o attention loss: the attention weights of the model trained without the auxiliary attention loss; (c) w/ attention loss: the attention weights with the attention loss.

6 Conclusions

In this paper, we adopt the table-filling approach for modeling the sentence-level correlation between two passages, and propose the attention-guided multi-layer multi-cross (MLMC) encoding scheme for the argument pair extraction (APE) task. Our model can better capture the internal relations between a review and its rebuttal with two sequence encoders and a table encoder via mutual attention mechanism. We also introduce an auxiliary attention loss to further improve the efficacy of the mutual attentions. Extensive experiments on the benchmark dataset demonstrate the effectiveness of our model architecture, which is potentially beneficial for other NLP tasks.

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*.

Elena Cabrio and Serena Villata. 2018. Five years of argument mining: a data-driven analysis. In *IJCAI*.

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

Liying Cheng, Lidong Bing, Qian Yu, Wei Lu, and Luo Si. 2020. Argument pair extraction from peer review and rebuttal via multi-task learning. In *Proceedings of EMNLP*.

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*.

Steffen Eger, Johannes Daxenberger, and Iryna Gurevych. 2017. Neural end-to-end learning for computational argumentation mining. In *Proceedings of ACL*.

Yufan Guo, Anna Korhonen, and Thierry Poibeau. 2011. A weakly-supervised approach to argumentative zoning of scientific documents. In *Proceedings of EMNLP*.

Pankaj Gupta, Hinrich Schütze, and Bernt Andrassy. 2016. Table filling multi-task recurrent neural network for joint entity and relation extraction. In *Proceedings of COLING*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*.

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

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*.

Makoto Miwa and Yutaka Sasaki. 2014. Modeling joint entity and relation extraction with table representation. In *Proceedings of EMNLP*.

Raquel Mochales Palau and Marie-Francine Moens. 2009. Argumentation mining: the detection, classification and structure of arguments in text. In *Proceedings of the 12th international conference on artificial intelligence and law*.

Isaac Persing and Vincent Ng. 2015. Modeling argument strength in student essays. In *Proceedings of ACL*.

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

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*.

Swapna Somasundaran, Brian Riordan, Binod Gyawali, and Su-Youn Yoon. 2016. Evaluating argumentative and narrative essays using graphs. In *Proceedings of COLING*.

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

Christian Stab and Iryna Gurevych. 2017. Parsing argumentation structures in persuasive essays. *Computational Linguistics*.

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

Simone Teufel, Advaith Siddharthan, and Colin Batchelor. 2009. Towards domain-independent argumentative zoning: Evidence from chemistry and computational linguistics. In *Proceedings of EMNLP*.

Jue Wang and Wei Lu. 2020. Two are better than one: Joint entity and relation extraction with table-sequence encoders. In *Proceedings of EMNLP*.

Zhen Wu, Chengcan Ying, Fei Zhao, Zhifang Fan, Xinyu Dai, and Rui Xia. 2020. Grid tagging scheme for aspect-oriented fine-grained opinion extraction. In *Findings of EMNLP*.

Fan Zhang and Diane Litman. 2016. Using context to predict the purpose of argumentative writing revisions. In *Proceedings of NAACL*.

Meishan Zhang, Yue Zhang, and Guohong Fu. 2017. End-to-end neural relation extraction with global optimization. In *Proceedings of EMNLP*.

A Details of Argument Pair Predictor

A.1 Argument Predictor

We cast the task of predicting argument spans as a sequence labeling problem. We adopt conditional random field (CRF) (Lafferty et al., 2001) that assigns each label sequence a score. The probability for each sequence (both review and reply) is defined as following:

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

where $\mathbf{s}$ represents the original sequence and $\mathbf{y}$ is the gold sequence label encoding argument spans under the BIO scheme (Ramshaw, 1995; Ratinov and Roth, 2009). The *score* function is defined as:

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

A is the matrix with trainable parameters representing transition scores within the CRF layer and F_{θ_1} represents the emission scores obtained after feeding review sequence and rebuttal sequence into the multi-cross encoder with parameters θ_1. The negative log-likelihood loss for both review and reply sequence in each instance is then defined as:

$$\mathcal{L}_{seq}(A, \theta_1) = -\big(\log p(\mathbf{y_{rv}}|\mathbf{s_{rv}}) + \log p(\mathbf{y_{rb}}|\mathbf{s_{rb}})\big).$$

A.2 Pair Predictor

Given table features $\mathbf{T^{(k)}}$ and an MLP layer, the probability that two sentences are from an argument pair can be expressed as following:

$$p(\mathbf{y}^{pair} = \mathbf{1}|(\mathbf{s_{rv}}, \mathbf{s_{rb}})) = \frac{1}{1+\exp(-F_{\theta_2}(\mathbf{T^{(k)}}))}.$$

F_{θ_2} is a composite function of Linear and ReLU functions, the final linear function among which has an output dimension of 1. The pairing loss $\mathcal{L}_{pair}(\theta_2, \theta_1)$ for each instance is then defined as:

$$\mathcal{L}_{pair}(\theta_2, \theta_1) = -\sum_{i,j} \Big(y_{i,j}^{pair} \log p(y_{i,j}^{pair} = 1|\mathbf{s_{rv}}, \mathbf{s_{rb}}) + (1 - y_{i,j}^{pair}) \log(1 - p(y_{i,j}^{pair} = 1|\mathbf{s_{rv}}, \mathbf{s_{rb}}))\Big),$$

where θ_2 are parameters within the MLP layer.

Note that the attention loss is a function of θ_1 and the overall loss is a function of θ_1, A and θ_2. The formulae provided in the main paper omit the related parameters for brevity.

B More Experimental Details

B.1 Hyperparameters

We manually tune the hyperparameter values (e.g. weight for pair loss λ_1, weight for attention loss λ_2, decaying parameter γ of exponential moving average) for our proposed multi-layer multi-cross model. We manually tune the weight for pair loss λ_1 from 0.3 to 0.7 with step size of 0.1 (Table 4), the weight for attention loss λ_2 from 0.5 to 2.5 with step size of 0.5 (Table 5) and the decaying paramter γ of exponential moving average from 0.7 to 1 with step size of 0.1 (Table 6) for our proposed multi-layer multi-cross model. We select the best hyperparameters based on the best F_1 score achieved on the development set and apply them to the test set for evaluation. Specifically, λ_1 is set as 0.5, λ_2 is set as 2, γ is set as 0.9.

B.2 Running Time, Number of Parameters and Results on Development Set

Table 8 shows the running time, the number of parameters, and the results on the development set of our models on RR-Submission-v1 dataset. For the bi-cross models, as review sentences and rebuttal sentences are concatenated as one sequence in one sequence encoder during training, the sequences are generally longer. Thus, the bi-cross models require a longer running time. As the number of layers increases, the performance on the development set improves yet the performance on the test set becomes worse. It is plausibly because that the model might face the overfitting issue.

B.3 MLP v.s. CNN

We replace the MLP module with convolutional neural networks (CNN) to predict the pairs and compare their performance on RR-Submission-v1 dataset. The comparison results are presented in Table 7. The theoretical advantage of CNN over MLP is that CNN is able to capture surrounding information with the help of kernels. However, the experiment results show that the convolutional structure performs worse than the simple MLP structure. By examining the kernel weight of the convolution layers, we observe no significant magnitude difference between the center weights and the peripheral weights. Take a 3x3 kernel as an example, the center weights are the weights at the center grid, while the weights located in the rest of the 8 grids are peripheral weights. This indicates that CNN accords way more importance to the surrounding information (8 times more important in the case of a 3x3 kernel) than to the original grid. The over-emphasis on surrounding information brings too much noise to the pair prediction.

Models	Argument Mining			Sentence Pairing			APE		
	Prec.	Rec.	F_1	Prec.	Rec.	F_1	Prec.	Rec.	F_1
λ_1=0.3	68.47	70.67	69.55	59.09	43.05	49.81	38.06	24.94	30.14
λ_1=0.4	68.50	70.67	69.57	60.02	46.07	52.13	**39.13**	26.99	31.95
λ_1=0.5	69.52	70.03	69.77	60.81	47.14	53.11	38.70	**27.93**	**32.44**
λ_1=0.6	66.56	71.87	69.12	59.32	44.93	51.13	36.59	26.27	30.59
λ_1=0.7	68.62	70.20	69.40	57.39	46.39	51.31	36.44	27.27	31.19

Table 4: Performance of Multi-cross (n=3) when applying different pair loss weights λ_1.

Models	Argument Mining			Sentence Pairing			APE		
	Prec.	Rec.	F_1	Prec.	Rec.	F_1	Prec.	Rec.	F_1
λ_2=0.5	67.68	70.20	68.92	58.28	45.82	51.30	35.20	26.83	30.45
λ_2=1	68.83	70.18	69.50	62.39	44.86	52.20	**40.23**	26.99	32.31
λ_2=1.5	67.83	71.01	69.39	57.26	46.45	51.29	35.52	27.88	31.24
λ_2=2	69.52	70.03	69.77	60.81	47.14	53.11	38.70	**27.93**	**32.44**
λ_2=2.5	66.60	70.72	68.60	60.15	44.72	51.30	36.64	26.33	30.64

Table 5: Performance of Multi-cross (n=3) when applying different attention loss weights λ_2.

Models	Argument Mining			Sentence Pairing			APE		
	Prec.	Rec.	F_1	Prec.	Rec.	F_1	Prec.	Rec.	F_1
γ=1	67.40	71.14	69.22	60.09	45.19	51.59	37.76	25.77	30.64
γ=0.9	69.52	70.03	69.77	60.81	47.14	53.11	**38.70**	**27.93**	**32.44**
γ=0.8	66.17	70.77	68.39	58.92	47.46	52.58	34.12	26.44	29.79
γ=0.7	67.40	70.18	68.76	60.21	46.43	52.43	36.36	26.11	30.39

Table 6: Performance of Multi-cross (n=3) when applying different values for the decaying parameter γ of exponential moving average.

Models	Argument Mining			Sentence Pairing			APE		
	Prec.	Rec.	F_1	Prec.	Rec.	F_1	Prec.	Rec.	F_1
with CNN	65.91	71.68	68.67	59.56	44.01	50.62	36.77	26.44	30.76
with MLP	69.52	70.03	69.77	60.81	47.14	53.11	**38.70**	**27.93**	**32.44**

Table 7: Performance of Multi-cross (n=3) when adopting CNN or MLP for pair prediction.

Models	RT (min)	# Params	APE F_1 (Dev)
Bi-Cross (n=1)	33	5.7M	36.27
Bi-Cross (n=2)	61	7.4M	37.23
Bi-Cross (n=3)	86	9.1M	37.16
Bi-Cross (n=4)	104	10.8M	37.11
Bi-Cross (n=5)	115	12.5M	38.35
Multi-Cross (n=1)	22	5.8M	35.07
Multi-Cross (n=2)	33	7.4M	36.60
Multi-Cross (n=3)	44	9.1M	36.65
Multi-Cross (n=4)	55	10.8M	35.26
Multi-Cross (n=5)	66	12.5M	36.72

Table 8: Running Time (RT) per epoch (minutes), number of parameters and the APE F_1 score of our models on RR-Submission-v1 dataset.

C More Experimental Analysis

C.1 Case Study on Attention Weights

Each row in Table 9 in which the exact gold and predicted results are shown corresponds to the respective row in Figure 4 in the main paper. We can see that instance (1) and instance (2) are perfectly predicted whereas one predicted reply argument is shorter than the gold argument in instance (3) and some argument pairs are identified wrongly in instance (4).

Attention distribution turns out to be strongly connected with the final output of the model, as attention weights exhibit exactly the same error as the wrongly predicted argument spans and argument pairs. In instance (3), we can see from the attention visualization that the review argument at position 15 only refers to the reply sentences from position 14 to 16. The wrong prediction of reply span (14, 16) (gold: (14, 26)) directly results from the inaccurate distribution of attention weights. For instance in Figure 4 row (4) as highlighted in red, it can also be noticed that some review arguments

Sample	Gold Review Arguments	Pred Review Arguments	Gold Rebuttal Arguments	Pred Rebuttal Arguments	Gold Argument Pairs	Pred Argument Pairs
(1)	(7,9)	(7,9)	(2,10)	(2,10)	(7,9) - (2,10)	(7,9) - (2,10)
	(10,12)	(10,12)	(12,14)	(12,14)	(10,12) - (12,14)	(10,12) - (12,14)
	(13,13)	(13,13)	(16,24)	(16,24)	(13,13) - (16,24)	(13,13) - (16,24)
(2)	(7,7)	(7,7)	(3,4)	(3,4)	(7,7) - (3,4)	(7,7) - (3,4)
	(8,11)	(8,11)	(6,8)	(6,8)	(8,11) - (5,8)	(8,11) - (6,8)
	(12,14)	(12,14)	(10,12)	(10,12)	(12,14) - (10,12)	(12,14) - (10,12)
	(15,17)	(15,17)	(14,18)	(14,18)	(15,17) - (14,18)	(15,17) - (14,18)
(3)	(9,10)	(9,10)	(4,5)	(4,5)	(9,10) - (4,5)	(9,10) - (4,5)
	(11,13)	(11,13)	(7,9)	(7,9)	(11,13) - (7,9)	(11,13) - (7,9)
	(14,15)	(14,15)	(12,13)	(12,13)	(14,15) - (12,13)	(14,15) - (12,13)
	(16,16)	(16,16)	(14,26)	(14,16)	(16,16) - (14,26)	(16,16) - (14,16)
	(17,18)	(17,18)	(29,31)	(29,31)	(17,18) - (29,31)	(17,18) - (29,31)
	(19,19)	(19,19)	(32,33)	(32,33)	(19,19) - (32,33)	(19,19) - (32,33)
	(20,21)	(20,21)	(36,38)	(36,38)	(20,21) - (36,38)	(20,21) - (36,38)
(4)	(8,9)	(8,9)	(2,5)	(2,7)	(8,9) - (2,5)	(8,9) - (2,7)
	(10,12)	(10,12)	(6,7)		(10,12) - (6,7)	(10,12) - (8,9)
	(13,13)	(13,13)	(8,9)	(8,9)	(13,13) - (8,9)	(13,13) - (10,10)
	(14,14)	(14,14)	(10,10)	(10,10)	(14,14) - (10,10)	
	(15,15)	(15,15)	(11,12)	(11,12)	(15,15) - (11,12)	(15,15) - (11,12)
	(16,16)	(16,16)	(13,16)	(13,16)	(16,16) - (13,16)	(16,16) - (13,16)
	(17,17)	(17,17)	(17,18)	(17,18)	(17,17) - (17,18)	(17,17) - (17,18)
	(18,18)	(18,18)				
	(19,20)	(19,20)	(20,21)	(20,21)	(19,20) - (20,21)	(19,20) - (20,21)
	(21,21)	(21,21)	(22,22)	(22,22)	(21,21) - (22,22)	(21,21) - (22,22)

Table 9: Gold and predicted review arguments, rebuttal arguments and argument pairs for four test data samples.

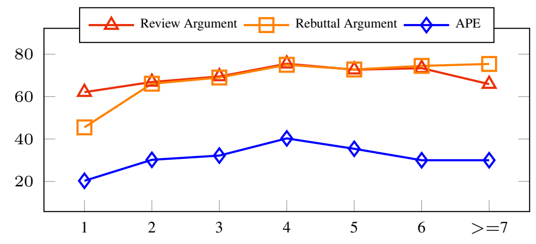


Figure 5: F_1(%) scores (y-axis) with different number of argument pairs (x-axis).

attend to the wrong rebuttal argument and some rebuttal arguments attend to the wrong review argument. The attention blocks in Figure 4 row (4) are (8, 9) - (2, 7), (10, 12) - (8, 9), (13, 13) - (10, 10) and the wrongly predicted argument pairs are also (8, 9) - (2, 7), (10, 12) - (8, 9), (13, 13) - (10, 10). Together with all 4 test instances, the conclusion can be reached that one-to-one correspondence can be found in the predicted paired arguments and the distribution of attention weights. Therefore, the hindrance to further improve the model performance comes from the inaccurately allocated attention weights.

C.2 Breakdown by Argument Density

We further evaluate the multi-cross (n=3) model performance on RR-Submission-v1 among different numbers of argument pairs in each instance. Figure 5 shows the argument mining performance on review and rebuttal separately and the overall APE performance. Their F_1 scores all increase as the number of argument pairs grows from 1 to 4 and reach plateaus afterwards. The reason is likely to be that most of the review and rebuttal pairs with about 4 argument pairs are written in a more formatted manner and are hence easier to be extracted. When the number of argument pairs is smaller than 3, it is highly likely that authors only reply to one or two review arguments. The irregular format might increase the difficulty of pair extraction. When the number of argument pairs is larger than average, the F_1 score of APE decreases slightly as the structure becomes more complicated.

In addition, we can see from Figure 5 that when the number of argument pairs is from 2 to 6, the F_1 scores of the argument mining subtask between review and rebuttal are very close. Compare to the multi-task model in the previous work (Cheng et al., 2020), our model's performance on the argument mining subtask between review and rebuttal is more balanced, which leads to the better overall APE performance.