Probing Graph Decomposition for Argument Pair Extraction

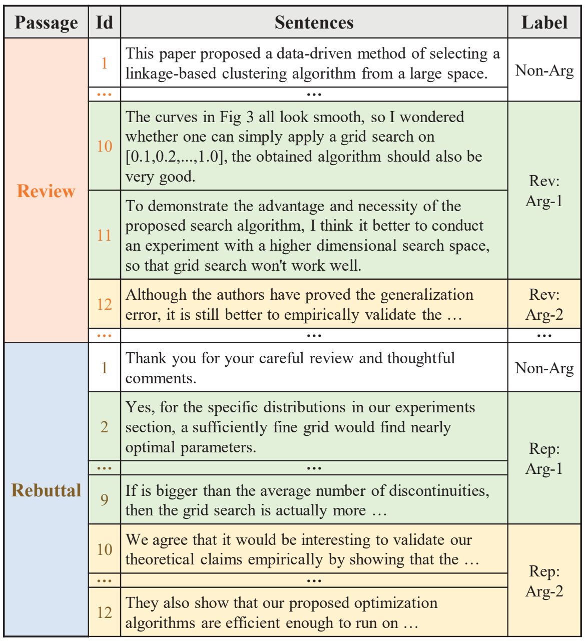
Yang Sun[1,2], **Bin Liang**[1,4], **Jianzhu Bao**[1,2], **Yice Zhang**[1,2], **Geng Tu**[1,4]
Min Yang[3*] and **Ruifeng Xu**[1,2,4*]
[1] Harbin Institute of Technology, Shenzhen, China [2] Peng Cheng Laboratory, Shenzhen, China
[3] SIAT, Chinese Academy of Sciences, Shenzhen, China
[4] Guangdong Provincial Key Laboratory of Novel Security Intelligence Technologies
sy95@mail.ustc.edu.cn, bin.liang@stu.hit.edu.cn
jianzhubao@gmail.com, 22b951011@stu.hit.edu.cn
zhangyc_hit@163.com, min.yang@siat.ac.cn, xuruifeng@hit.edu.cn

Abstract

Argument pair extraction (APE) aims to extract interactive argument pairs from two passages within a discussion. The key challenge of APE is to effectively capture the complex context-aware interactive relations of arguments between the two passages. In this paper, we elicit relational semantic knowledge from large-scale pre-trained language models (PLMs) via a probing technique. The induced sentence-level relational probing graph can help capture rich explicit interactive relations between argument pairs effectively. Since the relevance score of a sentence pair within a passage is generally larger than that of the sentence pair from different passages, each sentence would prefer to propagate information within the same passage and under-explore the interactive relations between two passages. To tackle this issue, we propose a graph decomposition method to decompose the probing graph into four subgraphs from intra- and inter-passage perspectives, where the intra-passage graphs can help detect argument spans within each passage and the inter-passage graphs can help identify the argument pairs between the review and rebuttal passages. Experimental results on two benchmark datasets show that our method achieves substantial improvements over strong baselines for APE.

1 Introduction

Dialogical argumentation, which focuses on the analysis of argumentation in debates or discussions, has rapidly emerged as a hot research topic in recent years. Argument pair extraction (APE) (Cheng et al., 2020) is a new and challenging task in the field of dialogical argumentation, which aims to extract interactive argument pairs from two argumentative passages within a discussion. As illustrated in Figure 1, a peer review process involves rich interactive arguments with each argument consisting of several consecutive sentences. An argument in the review can form an argument pair with the corresponding argument in the rebuttal that discusses the same topic.



Passage	Id	Sentences	Label
Review	1	This paper proposed a data-driven method of selecting a linkage-based clustering algorithm from a large space.	Non-Arg
	…	…	
	10	The curves in Fig 3 all look smooth, so I wondered whether one can simply apply a grid search on [0.1,0.2,...,1.0], the obtained algorithm should also be very good.	Rev: Arg-1
	11	To demonstrate the advantage and necessity of the proposed search algorithm, I think it better to conduct an experiment with a higher dimensional search space, so that grid search won't work well.	
	12	Although the authors have proved the generalization error, it is still better to empirically validate the …	Rev: Arg-2
	…	…	…
Rebuttal	1	Thank you for your careful review and thoughtful comments.	Non-Arg
	2	Yes, for the specific distributions in our experiments section, a sufficiently fine grid would find nearly optimal parameters.	Rep: Arg-1
	…	…	
	9	If is bigger than the average number of discontinuities, then the grid search is actually more …	
	10	We agree that it would be interesting to validate our theoretical claims empirically by showing that the …	Rep: Arg-2
	…	…	
	12	They also show that our proposed optimization algorithms are efficient enough to run on …	

Figure 1: A example of APE where a review passage and its corresponding rebuttal passage are shown on the upper and bottom. Rev:Arg-i/Rep:Arg-i denotes the i-th argument in the review/rebuttal and forms the i-th paired arguments. The white area refers to non-argument, while the green and yellow areas refer to argument pairs.

The core of APE is to detect the arguments within each passage and construct the relations between interactive arguments in the two passages. Most existing works (Cheng et al., 2020, 2021; Bao et al., 2022) apply powerful encoders, such as table encoders and attention mechanisms, to learn the sentence-level semantic representation for implicitly modeling relationships between argument pairs. However, as revealed in (Cheng et al., 2021), the sentence representations learned by pure attention-based methods are difficult to effectively capture

[*] Min Yang and Ruifeng Xu are corresponding authors.

Findings of the Association for Computational Linguistics: ACL 2023, pages 13075–13088
July 9-14, 2023 ©2023 Association for Computational Linguistics

the complicated relations between the sentences from different passages. Bao et al. (2021) explicitly established argument links based on co-occurring words within the sentence pairs and verified the importance of word-level relations among arguments. Nevertheless, they ignore the fact that the sentence pairs within the argument pairs generally contain semantically similar words, such as "space" and "interval" in the sentence pair, as illustrated in Figure 2. Although the semantic-aware relational information is already contained in the continuous representations of PLMs, neural networks lack an optimal mechanism to benefit from such information.

To address the aforementioned issues, we propose a novel ProbIng Graph dEcompositiON (PIGEON) framework for argument pair extraction by exploiting explicit semantic knowledge induced from large-scale PLMs. Specifically, we employ a two-stage masked language modeling process to construct sentence-level relation graphs between sentences (global linguistic properties) based on the number of highly similar word pairs within each sentence pair. The key idea behind this probing method is that we can obtain similar sentence pair representations when we mask out one word from a word pair with high similarity. The sentence-level relation graph is essential to effectively identify argument pairs.

Since the review and rebuttal passages have different writing styles and word distributions, the learned sentence-level probing graph may under-explore the interactive relations between the two passages. In particular, the relevance score of each sentence pair within a passage is generally larger than sentence pairs from different passages. For example, as shown in Figure 1, the tenth review sentence contains more semantically similar words with the eleventh review sentence than the second rebuttal sentence[1]. Consequently, the review sentences would prefer to propagate information within the same passage and under-explore the interactive relation between two passages. To effectively capture argument relations of different passages, we decompose the sentence-level probing graph into four sub-graphs from intra- and inter-passage perspectives. The intra-passage graph can help detect the argument spans within each passage, while the inter-passage graph is used to identify the argument pairs from different passages. To further improve the performance of our method, we also design an auxiliary graph contrastive loss to weaken the impact of noisy edges brought by the probing procedure.

Our contributions can be summarized as follows. (1) We propose a probing technique to elicit semantic-aware relational knowledge from large-scale PLMs for constructing sentence-level probing graph. (2) We decompose the sentence-level probing graph into four sub-graphs from intra- and inter-passage perspectives so as to effectively detect argument spans within each passage via intra-passage graphs and identify argument pairs from two passages via inter-passage graphs. (3) We conduct experiments on two APE benchmark datasets, and the results show that our method outperforms the strong baselines by a noticeable margin.

2 Methodology

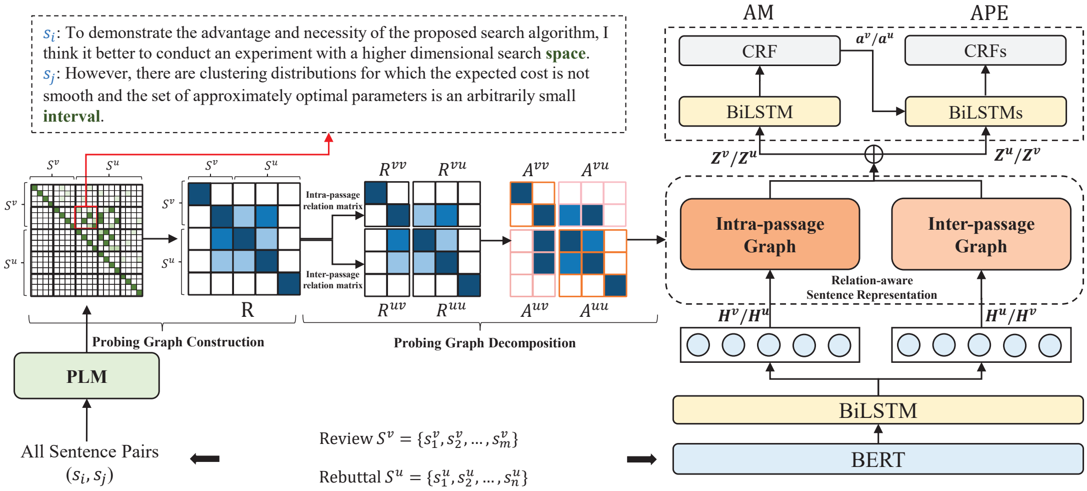
Following previous works (Cheng et al., 2020; Bao et al., 2021), we aim to automatically extract interactive argument pairs from the review and rebuttal passages by casting the argument mining and argument pairs extraction as two sentence-level sequence labeling problems using the standard BIOES scheme (Ratinov and Roth, 2009). Formally, given a review passage $S^v = \{s_1^v, \dots, s_m^v\}$ consisting of m sentences and a rebuttal passage $S^u = \{s_1^u, \dots, s_n^u\}$ consisting of n sentences, we first identify argument spans within the review and rebuttal passages, and obtain a review argument spans set $X^v = \{x_1^v, x_2^v, \dots\}$ and a rebuttal argument spans set $X^u = \{x_1^u, x_2^u, \dots\}$, where x_i^v and x_j^u are sentence-level spans in review and rebuttal passages, respectively. Then, we extract the paired arguments from review and rebuttal passages, and a set of interactive argument pairs $P = \{p_1, p_2, \dots\}$ can be collected, where $p = (x_i^v, x_j^u)$ is an interactive argument pair.

As illustrated in Figure 2, PIGEON contains four components, including a sentence representation learning module, a probing graph construction module, a graph decomposition module, and an argument pair prediction module. Next, we will describe each component of PIGEON in detail.

2.1 Sentence Representation Learning

We first apply BERT (Devlin et al., 2018) to obtain the word-level context representation of each sentence within the review and rebuttal passages. Then, we use a shared bidirectional LSTM (BiL-

[1] Further analysis is presented in Appendix A.1



s_i: To demonstrate the advantage and necessity of the proposed search algorithm, I think it better to conduct an experiment with a higher dimensional search **space**.
s_j: However, there are clustering distributions for which the expected cost is not smooth and the set of approximately optimal parameters is an arbitrarily small **interval**.
S^v
S^u
R
Intra-passage relation matrix
Inter-passage relation matrix
R^{vv}
R^{vu}
R^{uv}
R^{uu}
A^{vv}
A^{vu}
A^{uv}
A^{uu}
Probing Graph Construction
Probing Graph Decomposition
PLM
All Sentence Pairs
(s_i, s_j)
Review $S^v = \{s_1^v, s_2^v, \ldots, s_m^v\}$
Rebuttal $S^u = \{s_1^u, s_2^u, \ldots, s_n^u\}$
AM
APE
CRF
CRFs
a^v/a^u
BiLSTM
BiLSTMs
Z^v/Z^u
Z^u/Z^v
Intra-passage Graph
Inter-passage Graph
Relation-aware Sentence Representation
H^v/H^u
H^u/H^v
BiLSTM
BERT

Figure 2: The architecture of PIGEON.

STM) (Hochreiter and Schmidhuber, 1997) to encode the sentence-level dependencies within each passage. Specifically, we feed each sentence s_i into BERT and obtain the sentence embedding $\mathbf{e}_i \in \mathbb{R}^{d_b}$ by mean pooling over all token representations, where d_b is the vector dimension of the last layer of BERT. A BiLSTM is then utilized to encode the sentence representations $\{\mathbf{e}_1, \mathbf{e}_2, \ldots\}$ within each passage into the contextual sentence representations $\{\mathbf{h}_1, \mathbf{h}_2, \ldots\}$, where $\mathbf{h}_i \in \mathbb{R}^d$ and d is the hidden size of BiLSTM. We denote the contextual representations of review and rebuttal as $H^v = \{\mathbf{h}_1^v, \mathbf{h}_2^v, \ldots, \mathbf{h}_m^v\}$ and $H^u = \{\mathbf{h}_1^u, \mathbf{h}_2^u, \ldots, \mathbf{h}_n^u\}$, respectively.

2.2 Probing Graph Construction

To effectively capture the relation between each sentence pair, we need to detect the semantically similar words from the sentence pair. So far, there are many possible ways to derive semantically similar words, such as WordNet (Miller, 1995), Word2Vec (Mikolov et al., 2013), and GloVe (Pennington et al., 2014). However, these methods generally focus on context-free word similarity and ignore the context, failing to deal with words with multiple meanings in different contextual scenarios. In this paper, we elicit explicit semantic knowledge from large-scale pre-trained language models (PLMs) and build a relation graph between each sentence pair through a probing procedure. It is worth noting that such semantic knowledge is extracted in an unsupervised and off-the-line manner.

Formally, given a sentence pair (s_i, s_j), we first concatenate (s_i, s_j) and obtain a single sequence $s = \{\texttt{[CLS]}, s_i, \texttt{[SEP]}, s_j, \texttt{[SEP]}\}$, where [CLS] and [SEP] represent the classification and separation tokens respectively. Then, we propose a probing approach with a masking technique to learn the semantic similarity between arbitrary word pairs from each sentence pair. We employ a two-stage masked language modeling (MLM) process to measure the impact a context word has on predicting another word. After the probing process, we can construct sentence-level relation graphs based on the number of highly similar word pairs with each sentence pair. The key idea behind the probing process is that we can obtain similar sentence pairs representations when we mask out one word from a word pair with high similarity. Concretely, we replace the k-th token w_k^i in sentence s_i with a special mask token [mask] and feed the obtained new sequence $s_{/w_k^i}$ into BERT. We can obtain the contextualized representation $\mathbf{h}_{/w_k^i}$ of the k-th token. To calculate the correlation between w_k^i and the t-th word w_t^j in sentence s_j, we further mask out w_t^j from $s_{/w_k^i}$ to obtain the second corrupted sequence $s_{/w_k^i w_t^j}$ and feed it into BERT. We use $\mathbf{h}_{/w_k^i w_t^j}$ to denote the new representation of word w_k^i when both w_k^i and w_t^j are masked out simultaneously.

After that, we measure the distance $f(w_k^i, w_t^j)$ between $\mathbf{h}_{/w_k^i}$ and $\mathbf{h}_{/w_k^i w_t^j}$ to induce the semantically correlation between the k-th word w_k^i of s_i and the t-th word w_t^j of s_j. Here, we use the Euclidean distance metric to implement the distance function $f(\cdot)$ due to its simplicity and effectiveness as follows:

$$f(w_k^i, w_t^j) = ||\mathbf{h}_{/w_k^i} - \mathbf{h}_{/w_k^i w_t^j}||_2 \qquad (1)$$

Note that one word may be split into multiple tokens, we mask all tokens for each split-up word and apply a mean pooling over the token representations to obtain the word representation.

Word-level Similarity Matrix By repeating the two-stage MLM process on each pair of words (w_k^i, w_t^j) of the sentence pair (s_i, s_j), we can obtain a word-level similarity graph $M = \{M_{k,t}\}_{k=1,t=1}^{|s_1|,|s_2|}$ for the sentence pair (s_i, s_j), where $M_{k,t} = f(w_k^i, w_t^j)$ denotes the relation between word pair (w_k^i, w_t^j). Then, we utilize the min-max normalization to reduce the impact of the range of correlation scores. The normalized word-level similarity matrix $\hat{M}_{k,t}$ is computed by:

$$\hat{M}_{k,t} = \frac{M_{k,t} - min}{max - min} \tag{2}$$

where max and min is the maximum and minimum similarity scores of all word pairs in the review and rebuttal passages.

Sentence-level Probing Graph We construct the sentence-level probing graph in which nodes are sentences. The sentence-level relation matrix can be derived from the word-level relation matrices of all sentence pairs using the probing procedure. Specifically, we compute the relevance between each sentence pair (s_i, s_j) based on the word pairs with high semantic similarity. Formally, we compute the relation between the sentence pair (s_i, s_j) as follows:

$$R_{i,j} = \sum_{k=1}^{|s_i|} \sum_{t=1}^{|s_j|} I(\hat{M}_{k,t} > \sigma) \tag{3}$$

where σ is a pre-defined threshold. I is an indicator function. By traversing all sentence pairs, we can obtain the symmetrical sentence-level relation matrix $R = \{R_{i,j}\}_{i=1,j=1}^{|m+n|,|m+n|}$ for the review and rebuttal passages. Intuitively, if two sentences have many semantically similar word pairs, the corresponding edge will have a large weight.

2.3 Graph Decomposition

The review and rebuttal passages have different writing styles and word distributions. The relevance score of each sentence pair within a passage is generally larger than the sentence pair from different passages. To effectively capture argument relations of different passages, we decompose the sentence-level probing graph into four sub-graphs from intra- and inter-passage perspectives. The intra-passage graph can help detect the argument spans within the review (or rebuttal) passage, while the inter-passage graph is used to identify the argument pairs from the review and rebuttal passages. Formally, we decompose the sentence-level relation matrix $R \in \mathbb{R}^{(m+n)\times(m+n)}$ of two passages (S^v, S^u) into four sub-matrices $R^{vv} \in \mathbb{R}^{m\times m}$, $R^{uu} \in \mathbb{R}^{n\times n}$, $R^{vu} \in \mathbb{R}^{m\times n}$ and $R^{uv} \in \mathbb{R}^{n\times m}$, as illustrated in the left part of Figure 2. Among these, R^{vv} and R^{uu} represent the intra-passage relation matrices of the review and rebuttal passages, respectively. R^{vu} and R^{uv} denote the inter-passage relation matrices.

Intra-passage Graph Construction The intra-passage graph of each passage takes the sentences within the passage as vertices. The embeddings of the vertices are initialized with the corresponding sentence representations. Then, we refine the edges and the corresponding weights by the relative positions between sentences and intra-passage relevance matrices. Specifically, given a review passage S_v (or a rebuttal passage S_u), the edge weight $A_{i,j}^{vv}$ for each sentence pair (s_i, s_j) can be computed by:

$$A_{i,j}^{vv} = \begin{cases} \frac{R_{i,j}^{vv}}{max(R_i^{vv})} & if\ |i-j| \le \tau \\ 0 & otherwise \end{cases} \tag{4}$$

where τ is a pre-defined threshold. $max(R_i^{vv})$ represents the maximum value of the i-th row of intra-passage relation matrix R^{vv}.

Inter-passage Graph Construction The inter-passage graph for each passage is a bipartite graph, where each edge only exists between two sentences from different passages. The inter-passage adjacency matrix $A^{vu} \in \mathbb{R}^{m\times n}$ of passage S_v can be computed by:

$$A_{i,j}^{vu} = \frac{R_{i,j}^{vu}}{max(R^{vu})} \tag{5}$$

where $max(R^{vv})$ is the maximum value of the sentence-level relation matrix R^{vv}. Consequently, each passage can derive two different graphs (i.e. intra-passage and inter-passage graphs). Note that the intra- and inter-passage graphs of each passage are mutually independent though they are derived from the same passage.

Relation-aware Sentence Representations We use graph convolutional networks (GCNs) to learn the representations of intra- and inter-passage relation graphs. Each node in the graphs is updated according to the hidden representations of its neighborhoods based on the adjacency matrices of intra-passage and inter-passage graphs. Given the intra- and inter-passage graphs of passage S^v, the update of the l-th GCN block is defined as follows:

$$\begin{aligned} G^{vv} &= \mathrm{ReLU}(\hat{A}^{vv} Z^{v,l-1} W_1^l + b_1^l) \\ G^{vu} &= \mathrm{ReLU}(\hat{A}^{vu} Z^{u,l-1} W_2^l + b_2^l) \\ Z^{v,l} &= G^{vv} + G^{vu} + Z^{v,l-1} \end{aligned} \tag{6}$$

where $\hat{A}^{vv}$ (or $\hat{A}^{vu}$) is the normalized adjacency matrix learned from A^{vv} (or A^{vu}), and we have $\hat{A}_i^{vv} = \frac{A_i^{vv}}{\sum_j A_{i,j}^{vv}+1}$. $Z^{v,l-1}$ represents the sentence representations of the review passage evolved in the l-1-th GCN block. Here, the node representations of the first GCN layer are defined as $Z^{v,0} = H^v$. For simplicity, we denote the final output of the GCN blocks as Z^v. In this way, we can obtain the updated representation Z_i^v for the i-th sentence of passage S^v by integrating the representations of its neighbouring nodes within intra- and inter-passage graphs. Similarly, we can compute the updated sentence representations Z^u of passage S^u.

2.4 Argument Pair Prediction

After decomposing the probing graphs, the updated sentence representations are used for argument mining and argument pair extraction following previous works (Cheng et al., 2020; Bao et al., 2021).

Argument Mining We adopt a BiLSTM sequence tagger followed by a CRF sequence tagger to identify all potential arguments. Concretely, we feed the sentence representations Z^v and Z^u into the BiLSTM tagger to learn output hidden states $O^v = \{\mathbf{o}_1^v, \ldots, \mathbf{o}_m^v\}$ and $O^u = \{\mathbf{o}_1^u, \ldots, \mathbf{o}_n^u\}$. Then, O^v and O^u are put into the CRF tagger to predict the argument labels $\hat{Y}^v = \{\hat{y}_1^v, \ldots, \hat{y}_m^v\}$ and $\hat{Y}^u = \{\hat{y}_1^u, \ldots, \hat{y}_n^u\}$ for review and rebuttal respectively, where y_i is the BIOES label for the i-th sentence. Based on these two label sequences $\hat{Y}^v$ and $\hat{Y}^u$, we can further parse the potential argument spans for the review and rebuttal passages, i.e. $\hat{X}^v = \{\hat{x}_1^v, \hat{x}_2^v, \ldots\}$ and $\hat{X}^u = \{\hat{x}_1^u, \hat{x}_2^u, \ldots\}$, where $\hat{x}_i$ is the i-th predicted argument span. The sequence labeling loss $\mathcal{L}_{\mathrm{AM}}$ for each instance is defined as follows:

$$\mathcal{L}_{\mathrm{AM}} = -[\log p(\mathbf{Y}^v|\mathbf{o}^v) + \log p(\mathbf{Y}^u|\mathbf{o}^u)] \tag{7}$$

where $\mathbf{Y}^v$ and $\mathbf{Y}^v$ are the ground-truth sequence labels of review and rebuttal.

Argument Pair Extraction With the learned argument spans sets (X^v and X^b) and the argument-specific sentence representations ($\hat{Z}^v = Z^v + O^v$ and $\hat{Z}^u = Z^u + O^u$), we can extract argument pairs with dual sequence taggers. Specifically, we first produce the representation $a_k^v = \frac{1}{e_k-b_k+1}\sum_{i=b_k}^{e_k} \hat{Z}_i^v$ of the k-th extracted argument span $x_k^v = (b_k, e_k)$ by mean pooling over the corresponding argument-specific sentence representations. Then, we concatenate a_k^v to the argument-specific sentence representation $\hat{Z}^u$ of rebuttal S^u to obtain the argument x_k^v-specific sentence representations $\{[a_k^v, \hat{Z}_1^u], \ldots, [a_k^v, \hat{Z}_n^u]\}$, where $[\cdot, \cdot]$ is the concatenation operation. We feed the learned representations into a BiLSTM tagger and a CRF tagger to predict the argument label sequences $\mathbf{Y}_k^u$ with its paired arguments from the rebuttal passage. Similarly, we can perform the same procedure to capture its paired arguments from review for the k-th rebuttal argument by predicting the label sequences $\mathbf{Y}_k^v$ with another BiLSTM and CRF tagger. For APE, its sequence labeling loss $\mathcal{L}_{\mathrm{APE}}$ in each instance is defined as:

$$\mathcal{L}_{\mathrm{APE}} = -\sum_k \log p(\mathbf{Y}_k^u|a_k^v, \hat{Z}^u) - \sum_k \log p(\mathbf{Y}_k^v|a_k^u, \hat{Z}^v) \tag{8}$$

Graph Contrastive Loss We introduce an auxiliary graph contrastive loss to weaken the impact of the introduced noisy edges brought by the probing procedure. Taking the intra- and inter-passage graphs of review S^v as an example, we follow an i.i.d. uniform distribution to randomly drop the noisy edges (non-argument pairs) in the graph with probability μ and generate auxiliary graph views with adjacency matrices $\tilde{A}^{vv} = drop(\hat{A}^{vv})$ and $\tilde{A}^{vu} = drop(\hat{A}^{vu})$ from the original graph. Noted that the removal probabilities of the edges for the argument pairs are zero. Then, we feed the learn auxiliary graph views into GCNs and produce the auxiliary updated node representations $\tilde{Z}^v$ and $\tilde{Z}^u$ for the review S^v and rebuttal S^u respectively. After that, we employ a contrastive objective $\mathcal{L}_{\mathrm{GCL}}$ to distinguish the representations of different views of the same node from the representations of different nodes:

$$\mathcal{L}_{\mathrm{GCL}} = -\sum_i \log \frac{\exp(\psi(Z_i, \tilde{Z}_i))}{\exp(\sum_{j \neq i} \psi(Z_i, Z_j) + \sum_j \psi(Z_i, \tilde{Z}_j))} \tag{9}$$

where $Z = [Z^v, Z^u]$ represent the updated node representation matrices of passages S^v and S^u. $\psi(\cdot, \cdot)$ denotes the cosine similarity and $g(\cdot)$ is a two-layer perceptron.

Joint Training Objective We minimize the joint loss function $\mathcal{L}_{\text{joint}}$ by summing up the three training objectives as:

$$\mathcal{L}_{\text{joint}} = \mathcal{L}_{\text{AM}} + \mathcal{L}_{\text{APE}} + \lambda \mathcal{L}_{\text{GCL}} \tag{10}$$

where λ is a tuned hyper-parameter controlling the impact of $\mathcal{L}_{\text{GCL}}$.

3 Experimental Setup

Datasets We conduct experiments on the Review-Rebuttal (RR) dataset, which is a benchmark dataset proposed by (Cheng et al., 2020). The RR dataset includes 4,764 review-rebuttal pairs collected from ICLR 2013 to ICLR 2020. There are two versions provided: RR-Passage-v1 (RR-P) and RR-Submission-v2 (RR-S). Both RR-P and RR-S are split by the ratio of 8:1:1 into train, dev, and test sets. In the RR-P dataset, different review-rebuttal passage pairs of the same paper submission could be put into different sets, while in the RR-S dataset, multiple review-rebuttal passage pairs of the same submission are included in the same set. Since RR-S is more challenging than RR-P, we conduct further experiments on RR-S. The detailed statistics of RR-P and RR-S are summarized in Appendix A.2.

Evaluation Metrics Following previous works (Cheng et al., 2020; Bao et al., 2021), we adopt precision **(Prec.**), recall (**Rec.**) and $\mathbf{F_1}$ scores to measure the performance of both argument mining (AM) and argument pair extraction (APE).

Baselines To evaluate the effectiveness of PIGEON, we compare it with several strong baselines. **PL-H-LSTM-CRF** (Cheng et al., 2020) learns separate sequence labeling and sentence relation classification models, and then combines two results together to predict the argument pairs. Similar to PL-H-LSTM-CRF, **MT-H-LSTM-CRF** (Cheng et al., 2020) trains two subtasks via a shared feature encoder in the multi-task learning manner. **MLMC** (Cheng et al., 2021) is an attention-guided model based on a table-filling approach. **MGF** (Bao et al., 2021) proposes a mutual guidance framework with an inter-sentence relation graph. **MRC-APE** (Bao et al., 2022) applies machine reading comprehension framework with a Longformer (Beltagy et al., 2020) as the encoder, which is the state-of-the-art method on RR.

Implementation Details PIGEON is implemented in PyTorch on an NVIDIA TITAN RTX GPU. We apply the uncased BERT base[2] as our PLM. The AdamW optimizer (Loshchilov and Hutter, 2018) is employed for parameter optimization, and the initial learning rates for the BERT layer and other layers are set to 1e-5 and 1e-3, respectively. Similar to previous works (Cheng et al., 2021), we set the batch size as 1 due to the limited memory. The maximum number of the GCN blocks on RR-S and RR-P are set to 5 and 3, respectively. The hidden size of BiLSTMs is set to 256. In addition, the parameters of BiLSTMs and CRFs used in the three taggers are not shared [3]. All experiments are performed five times with different random seeds, and we report the averaged scores. Our code and data are available at `https://github.com/syiswell/PIGEON`.

4 Experimental Results

4.1 Overall Performance

We report the overall performance of our proposed framework and baseline methods in Table 1. Our method achieves the best performance on both RR-S and RR-P. On RR-S, our method outperforms the current state-of-the-art method (i.e., MRC-APE) by 2.94% in terms of F_1 score on the APE subtask. On RR-P, our model also exceeds MRC-APE and obtains about 3.05% higher F_1 score on the APE subtask. The experimental results verify the superiority of our method on the APE subtask. In addition, our PIGEON is also more efficient than baselines, as shown in Appendix A.4.

We also observe that the pipeline method (i.e., PLH-LSTM-CRF) perform more poorly than the other end-to-end baselines because it may lead to error propagation. The attention-based method (i.e., MLMC) achieves significant improvement over MT-H-LSTM-CRF, since MLMC can implicitly model the argument correlation. The graph-based method (i.e., MGF) surpasses MLMC but underperforms MRC-APE. This may be because MGF only considers the word overlap and explicitly constructs the incomplete argument correlation without considering semantic information. Our PIGEON performs better than all baselines by probing semantic knowledge from PLMs.

[2] We implement BERT using huggingface toolkit: https://huggingface.co/

[3] More details about hyperparameter settings can be found in Appendix A.3.

Dataset	Method	AM			APE		
		Pre.	Rec.	$\mathbf{F_1}$	Pre.	Rec.	$\mathbf{F_1}$
RR-S	PLH-LSTM-CRF	67.02	68.49	67.75	19.74	19.13	19.43
	MT-H-LSTM-CRF	70.74	69.46	70.09	27.24	26.00	26.61
	MLMC	69.53	**73.27**	71.35	37.15	29.38	32.81
	MGF	70.82	73.19	71.99	40.45	30.77	34.95
	MRC-APE	71.83	73.05	72.43	**41.83**	38.17	39.92
	PIGEON (Our)	**72.29**	73.22	**72.75**	41.06	**44.84**	**42.86**
RR-P	PLH-LSTM-CRF	73.10	67.65	70.27	21.24	19.30	20.22
	MT-H-LSTM-CRF	71.85	71.01	71.43	30.08	29.55	29.81
	MLMC	66.79	72.17	69.38	40.27	29.53	34.07
	MGF	73.62	70.88	72.22	38.03	35.68	36.82
	MRC-APE	**76.39**	70.62	73.39	37.70	44.00	40.61
	PIGEON (Our)	73.30	**73.56**	**73.43**	**43.18**	**44.16**	**43.66**

Table 1: Performance comparison. Our improvements over baselines are statistically significant with $p < 0.05$.

Dataset	Method	AM	APE
RR-S	PIGEON	72.75	42.86
	w/o GCL	72.16	42.33
	w/o Intra-PG	71.31	42.02
	w/o Inter-PG	71.77	35.45
	w/o PG	70.65	34.49
	w/o GD	71.71	39.81
RR-P	PIGEON	73.43	43.66
	w/o GCL	73.00	43.37
	w/o Intra-PG	72.23	42.39
	w/o Inter-PG	71.62	35.17
	w/o PG	70.49	34.48
	w/o GD	72.22	40.36

Table 2: Ablation test performance in terms of F_1.

Dataset	Method	AM	APE
RR-S	Co-occurrence	71.78	37.75
	Word2Vec	72.24	37.41
	Glove	72.13	37.73
	WordNet	72.15	38.32
	Probing	72.75	42.86
RR-P	Co-occurrence	71.91	38.47
	Word2Vec	72.15	38.49
	Glove	72.07	38.61
	WordNet	72.09	38.31
	Probing	73.43	43.66

Table 3: The F_1 scores when using different methods to detect semantically similar words.

4.2 Ablation Study

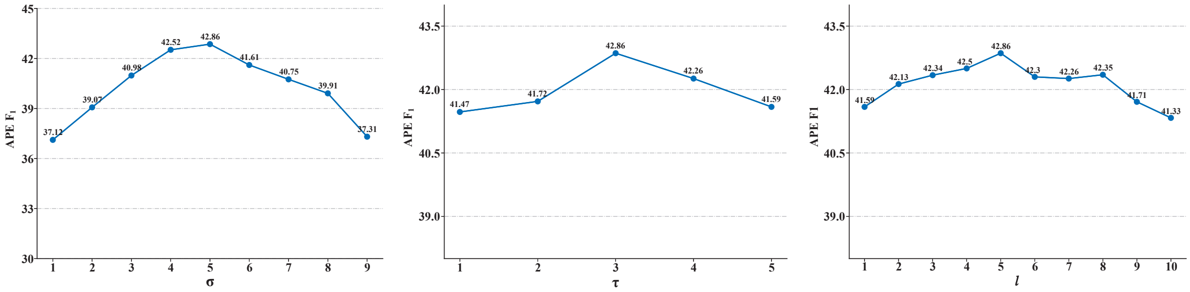
To analyze the impact of different components in our proposed PIGEON method, we conduct ablation test in terms of removing probing graph (w/o PG), removing graph decomposition (w/o GD), removing intra-passage graphs (w/o Intra-PG), removing inter-passage graphs (w/o Inter-PG), and removing graph contrastive learning (w/o GCL), respectively.The results are reported in Table 2. We can observe that both w/o Inter-PG and w/o Inter-PG degrade the performance substantially, verifying that the sentence relations within both the intra-passage and inter-passage graphs are important for the APE subtask. Removing graph decomposition (w/o GD) leads to performance drops significantly, demonstrating that probing graph decomposition can help alleviate the problem that the review and rebuttal passages have different styles and word distributions. It is no surprise that graph contrastive learning also contribute to the effectiveness of our method by reducing the impact of noisy edges in probing graphs.

4.3 Graph Parameters Analysis

We analyze the hyperparameters used in the processes of probing graph construction and decomposition, i.e., the threshold σ for determining word pairs with high semantic similarity, the distance threshold τ for constructing the intra-passage graph, and the number l of GCN blocks. The results on RR-S are illustrated in Figure 3. (i) From Figure 3a, we observe that the best results can be obtained when $\sigma = 0.5$. The small σ tend to introduce infrequent word pairs and lead to noisy argument relations. (ii) From Figure 3b, our method achieves the best performance when τ equals 3. A too large distance threshold may introduce incorrect edges in the graph, while a too small threshold will discard the correct relations between sentences. (iii) We vary the block number l in GCNs from 1 to 10, and illustrate the results in Figure 3c. We can achieve the best results when $l = 5$. While the performance tends to decrease as the number of GCN blocks increases when the block number is greater than 5. This implies that roughly increasing the number of GCN blocks is vulnerable to slash the learning ability of the model owing to the sharp increase of the model parameters.

4.4 Different Word Similarity Measures

We compare our probing method with several popular word similarity measures (e.g., Word2Vec,

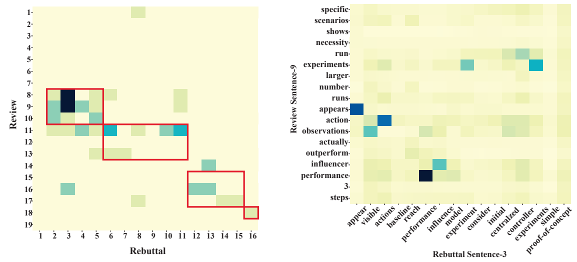


(a) Threshold

(b) Relative distance

(c) Graph layers

Figure 3: The impacts of graph parameters on RR-S.



(a)

(b)

Figure 4: The inter-passage adjacent matrix and a word-level similarity matrix of an example. The red blocks denote the ground-truth argument pairs. The word-level similarity matrix belongs to the sentence pair $(9, 3)$ of review and rebuttal passages.

GloVe and WordNet) for detecting word pairs with high semantic similarity. The co-occurrence based method focuses on string matching. The Word2Vec and GloVe methods obtain the word pair similarity by computing the cosine similarity of their word vectors. The WordNet method computes the word pair similarity based on the shortest path connecting the two word senses in a taxonomy. After learning the similarity of arbitrary word pairs, we construct the inter- and intra-passage graphs similar to our PIGEON method. We report the AM and APE results in Table 3. Our probing method performs significantly better than the compared word similarity measures by eliciting context-aware semantic knowledge from large-scale PLMs.

4.5 Case Study

We provide an exemplary case that is selected from RR-S test set by visualizing the adjacent matrix of the inter-passage graphs, where the distribution of edge weights is similar to the distribution of ground-truth argument pair labels. As shown in Figure 4b, the probing method can catch the semantic similarity between the word pair “observation” and “visible” with the help of elicited knowledge from PLMs. We believe that our PIGEON can probe rich semantic knowledge from PLMs, helping detect argument relations for APE.

5 Related Work

Argument Pair Extraction Most existing argument mining methods focus on modeling the arguments in monologues, such as argumentation structure parsing (Stab and Gurevych, 2014; Morio et al., 2020), argument quality assessment (Lauscher et al., 2020), and argumentation strategies modeling (Al Khatib et al., 2017). However, in real-life scenarios, arguments are often in the form of dialogues. Several prior studies detect agreement and disagreement in online debates and discussions (Morio and Fujita, 2018; Chakrabarty et al., 2019; Ji et al., 2021). Subsequently, Cheng et al. (2020) introduced a new argument pair extraction (APE) task in the domain of peer review and rebuttal, aiming to extract the argument pairs from the review and rebuttal passages simultaneously. Cheng et al. (2021) applied an attention mechanism and a table-filling approach to implicitly model the interaction between argument pairs. To explicitly model the relations between argument pairs, Bao et al. (2021) proposed a mutual guidance framework with an inter-sentence graph for APE. Bao et al. (2022) explored a bidirectional machine reading comprehension (MRC) to capture the interactions between argument pairs. Different from previous works, we explicitly capture the relations between argument pairs by eliciting context-aware semantic knowledge from PLM. In addition, we propose a graph decomposition method to deal with the issue that the review and rebuttal passages have different styles and word distribution.

Probing Knowledge from PLMs Recently, the success of PLMs has led to plenty of studies applying the probing technique to elicit rich knowledge from large-scale PLMs (Jawahar et al., 2019; Clark et al., 2019; Wu et al., 2020; Wang et al., 2022). A typical probing study is to investigate the knowledge and linguistic properties contained in PLMs, such as morphology (Belinkov et al., 2017), word sense (Reif et al., 2019), syntax (Hewitt and Manning, 2019; Dai et al., 2021). The key idea behind these works is to define a precise task, and then design a simple model (called a probe) to solve the task using the contextualized representations provided by PLMs. There are also some studies (Petroni et al., 2019; Zhong et al., 2021) that seek to answer to what extent PLMs store factual, relational and commonsense knowledge.

6 Conclusion

In this paper, we designed a probing technique to elicit semantic-aware relational knowledge from large-scale PLMs, which captured rich explicit interactive relations between argument pairs. In addition, we proposed a graph decomposition method to decompose the probing graph into four sub-graphs from intra- and inter-passage perspectives, which could alleviate the issue that different participants might have different writing styles and word distributions for the APE task. Experimental results on two benchmark datasets showed that our method outperformed strong baselines significantly.

Limitation

To better understand the limitations of the proposed model, we carry out an analysis of the errors made by PIGEON. Specifically, we randomly select 100 instances that are incorrectly predicted by PIGEON and summarize the primary types of error. The first error category is boundary prediction error. Since we modeled the APE task as a sequence labeling, our model may only recognize a part of an argument. Thus, multiple consecutive arguments may be identified as a single argument. The second type of error is caused by the absence of semantically similar words in the argument pairs. In this case, the proposed probing graphs cannot model the relations between argument pairs. Third, another error category occurs when semantically similar words are also presented in non-matching argument pairs. The argument relation may be misled by these words. It suggests that certain relation modeling method needs to be devised in the future so as to better infer argument relation. For example, we may leverage the high-level topic information over argument pairs to guide the learning of relation-specific features.

In addition, the proposed probing approach may be computationally expensive and we can alleviate this problem by saving the similarity of all word pairs for one time for the entire dataset. We will address this issue in future work.

Acknowledgements

We thank the anonymous reviewers for their valuable suggestions to improve the quality of this work. This work was partially supported by the National Natural Science Foundation of China (62006062, 62176076), Natural Science Foundation of GuangDong 2023A1515012922 the Shenzhen Foundational Research Funding (JCYJ20220818102415032), the Major Key Project of PCL2021A06, Guangdong Provincial Key Labo-ratory of Novel Security Intelligence Technologies 2022B1212010005. Min Yang was partially supported by the National Key Research and Development Program of China (2022YFF0902100), Shenzhen Science and Technology Innovation Program (KQTD20190929172835662), Shenzhen Basic Research Foundation (JCYJ20210324115614039 and JCYJ20200109113441941).

References

Khalid Al Khatib, Henning Wachsmuth, Matthias Hagen, and Benno Stein. 2017. Patterns of argumentation strategies across topics. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1351–1357.

Jianzhu Bao, Bin Liang, Jingyi Sun, Yice Zhang, Min Yang, and Ruifeng Xu. 2021. Argument pair extraction with mutual guidance and inter-sentence relation graph. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3923–3934.

Jianzhu Bao, Jingyi Sun, Qinglin Zhu, and Ruifeng Xu. 2022. Have my arguments been replied to? argument pair extraction as machine reading comprehension. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 29–35.

Yonatan Belinkov, Nadir Durrani, Fahim Dalvi, Hassan Sajjad, and James Glass. 2017. What do neural machine translation models learn about morphology?

In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 861–872, Vancouver, Canada. Association for Computational Linguistics.

Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*.

Tuhin Chakrabarty, Christopher Hidey, Smaranda Muresan, Kathleen Mckeown, and Alyssa Hwang. 2019. Ampersand: Argument mining for persuasive online discussions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2933–2943.

Liying Cheng, Lidong Bing, Qian Yu, Wei Lu, and Luo Si. 2020. Ape: argument pair extraction from peer review and rebuttal via multi-task learning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7000–7011.

Liying Cheng, Tianyu Wu, Lidong Bing, and Luo Si. 2021. Argument pair extraction via attention-guided multi-layer multi-cross encoding. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6341–6353.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D Manning. 2019. What does bert look at? an analysis of bert's attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 276–286.

Junqi Dai, Hang Yan, Tianxiang Sun, Pengfei Liu, and Xipeng Qiu. 2021. Does syntax matter? a strong baseline for aspect-based sentiment analysis with roberta. In *EMNLP*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

John Hewitt and Christopher D Manning. 2019. A structural probe for finding syntax in word representations. In *NAACL*.

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

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does bert learn about the structure of language? In *ACL 2019-57th Annual Meeting of the Association for Computational Linguistics*.

Lu Ji, Zhongyu Wei, Jing Li, Qi Zhang, and Xuan-Jing Huang. 2021. Discrete argument representation learning for interactive argument pair identification. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5467–5478.

Anne Lauscher, Lily Ng, Courtney Napoles, and Joel Tetreault. 2020. Rhetoric, logic, and dialectic: Advancing theory-based argument quality assessment in natural language processing. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4563–4574.

Ilya Loshchilov and Frank Hutter. 2018. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

George A Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

Gaku Morio and Katsuhide Fujita. 2018. End-to-end argument mining for discussion threads based on parallel constrained pointer architecture. In *Proceedings of the 5th Workshop on Argument Mining*, pages 11–21.

Gaku Morio, Hiroaki Ozaki, Terufumi Morishita, Yuta Koreeda, and Kohsuke Yanai. 2020. Towards better non-tree argument mining: Proposition-level biaffine parsing with task-specific parameterization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3259–3266.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473.

Lev Ratinov and Dan Roth. 2009. Design challenges and misconceptions in named entity recognition. In *Proceedings of the thirteenth conference on computational natural language learning (CoNLL-2009)*, pages 147–155.

Emily Reif, Ann Yuan, Martin Wattenberg, Fernanda B. Viégas, Andy Coenen, Adam Pearce, and Been Kim. 2019. Visualizing and measuring the geometry of BERT. In *NeurIPS*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Christian Stab and Iryna Gurevych. 2014. Annotating argument components and relations in persuasive essays. In *Proceedings of COLING 2014, the 25th international conference on computational linguistics: Technical papers*, pages 1501–1510.

Lihan Wang, Bowen Qin, Binyuan Hui, Bowen Li, Min Yang, Bailin Wang, Binhua Li, Jian Sun, Fei Huang, Luo Si, et al. 2022. Proton: Probing schema linking information from pre-trained language models for text-to-sql parsing. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 1889–1898.

Zhiyong Wu, Yun Chen, Ben Kao, and Qun Liu. 2020. Perturbed masking: Parameter-free probing for analyzing and interpreting bert. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4166–4176.

Zexuan Zhong, Dan Friedman, and Danqi Chen. 2021. Factual probing is [mask]: Learning vs. learning to recall. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5017–5033.

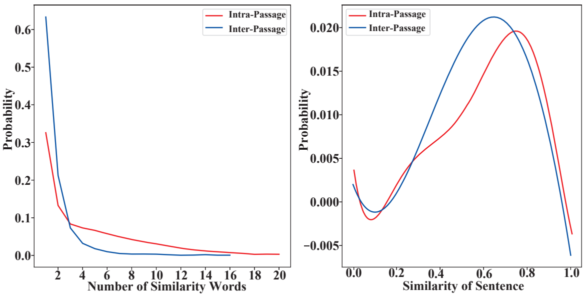
A Appendix

A.1 Sentence Relevance Analysis

In Figure 5, we randomly select 100 samples from the RR-S test set and show the distributions of the number of similar words and similarities of sentences from intra- and inter-passages. The number of similar words between sentences is computed by our probing procedure, while the sentence similarities are measured by the cosine similarity of sentence representations obtained from BERT. We can observe that the number of similar words in the sentences within a passage is larger than that in the sentences from different passages, as shown in Figure 5a. In addition, the sentence representations in the intra-passage, on average, have higher similarity than that in the inter-passage, as illustrated in Figure 5b. This may be because the review and rebuttal passages have different writing styles and word distributions.

A.2 Data Statistics

The detailed statistics of the RR-S and RR-P datasets are summarized in Table 4. Both RR-S and RR-P dataset contain 4,764 review-rebuttal pairs collected from ICLR 2013 to ICLR 2020, which are split by the ratio of 8:1:1 into train, dev, and test sets. In the RR-P dataset, different review-rebuttal passage pairs of the same paper submission could be put into different sets, while in the RR-S dataset, multiple review-rebuttal passage pairs of the same submission are included in the same set. Thus, RR-S is more challenging than RR-P.



(a) (b)

Figure 5: Visualizing the distributions of the number of similar words and similarities of sentences in the intra- and inter-passages.

Dataset	RR-S	RR-P
Train	3817	3811
Dev	473	477
Test	474	476
Sentences	217.8K	190.5K
Arguments	40.9K	40.5K
Argument Pairs	19.1K	18.6K
Avg. SPA in Review	2.84	2.53
Avg. SPA in rebuttal	4.16	3.82
Avg. SPA	3.41	3.09

Table 4: The statistics of the evaluated datasets, where SPA denotes sentences per argument

A.3 Hyperparameter Settings

We manually tune the hyperparameter values (e.g., the weight λ for graph contrastive loss and the drop probability μ of noisy edges) on the RR-S. We report the results in Table 5 and Table 6. The weight λ of graph contrastive loss is tuned from 0.001 to 1 with a ratio of 10. The drop probability μ is tuned from 0.1 to 0.5 with a step size of 0.1. From the results in Table 5 and Table 6, we set the value of λ to 0.01 and the value of μ to 0.1.

A.4 Computational Cost

Table 7 shows the training time, the testing time, the number of parameters, and the APE results of our model on the RR-S development set. As the number of GCN blocks increases, the performance on the development set improves yet the performance on the testing set becomes worse. It implies our model might suffer from the overfitting issue with the increasing layers of GCN blocks. In addition, our PIGEON is more efficient than baselines during inference owing to the fewer model parameters. Note that MRC-APE with fewer parameters

λ	AM	APE
1	71.53	41.99
0.1	71.81	42.32
0.01	72.75	42.86
0.001	72.19	41.94

Table 5: The F_1 scores when applying different values for the graph contrastive loss weight λ.

μ	AM	APE
0.1	72.75	42.86
0.2	72.27	42.03
0.3	72.19	41.45
0.4	72.31	41.83
0.5	71.94	42.17

Table 6: The F_1 scores when applying different values for the drop probability μ for noise edges

Model	Layer	RT (min)	TT (sec)	# Params	APE $\mathbf{F_1}$(Dev)
PIGEON	1	17.1	36	4.52M	44.10
	2	17.3	36	4.77M	44.50
	3	17.4	36	5.02M	44.77
	4	17.5	37	5.77M	44.23
	5	17.6	37	6.02M	45.02
	6	17.7	37	6.27M	44.68
	7	17.8	37	6.52M	44.58
	8	18.0	37	6.77M	45.07
	9	18.1	37	7.02M	44.46
	10	18.4	37	7.27M	45.00
MLMC	-	44	106	9.10M	36.65
MGF	-	15	48	7.27M	37.42
MRC-APE	-	100	188	0.94M	40.73

Table 7: Running Time (RT) per epoch (minutes), Testing Time (TT) in the test set (second), number of parameters (except the part of BERT or Longformer), and the APE F_1 score of our PIGEON with baselines on RR-S.

requires the most running and testing time because it needs to repeatedly implement multi-turn machine reading comprehension if there are several arguments within an instance.

Graph Augmentation	AM	APE
Noisy edge dropping	72.75	42.86
Random edge dropping	72.19	42.25

Table 8: The F_1 scores when applying different graph augmentation methods.

A.5 Different Graph Augmentation Methods

We also explore an additional random dropping method for graph augmentation. The results are provided in Table 8. The random dropping method randomly removes a certain percentage of edges, without considering whether the edge is a noisy edge or not. We find that the random dropping method performs worse than the noisy edge dropping method for graph contrastive learning. This may be because the random dropping method removes the important edges and corrupts the true argument relations. By contrast, the noisy edge dropping method only focuses on the noisy edges within non-argument pairs and achieves better performance.

Method	AM	APE
SBERT	70.93	35.83
SROBERTA	71.92	35.70
Probing	72.75	42.86

Table 9: The F_1 scores when applying different sentence similarity measures.

A.6 Different Sentence Similarity Measures

We also explore two sentence similarity measures i.e., SBERT and SROBERTA (Reimers and Gurevych, 2019). The results of SBERT, SROBERTA and our method on RR-S are presented in Table 9. We can observe that our method significantly outperforms both SBERT and SROBERT. The reason may be that sentences with a high degree of similarity are not certainly in the same argument pair, but may also be expressing similar opinions on different topics and then incorrectly establishing argument relations.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
Section Limitation

☒ A2. Did you discuss any potential risks of your work?
Our work does not have potential risks.

☑ A3. Do the abstract and introduction summarize the paper's main claims?
Section Abstract and Introduction

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☒ Did you use or create scientific artifacts?

Left blank.

□ B1. Did you cite the creators of artifacts you used?
No response.

□ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
No response.

□ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
No response.

□ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
No response.

□ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
No response.

□ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
No response.

C ☑ Did you run computational experiments?

section 4

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
section Appendix

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
section 3 and section Appendix

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
section 3

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
section 3

D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?
Left blank.

□ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
No response.

□ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
No response.

□ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
No response.

□ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
No response.

□ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
No response.