Discourse Structure-Aware Prefix for Generation-Based End-to-End Argumentation Mining

Yang Sun[1,2*], **Guanrong Chen**[1,6*], **Caihua Yang**[1,6], **Jianzhu Bao**[1,2], **Bin Liang**[1,4], **Xi Zeng**[3], **Min Yang**[5†] and **Ruifeng Xu**[1,2,6†]

[1] Harbin Institute of Technology, Shenzhen, China [2] Peng Cheng Laboratory, Shenzhen, China
[3] The 30th Research Institute of China Electronics Technology Group Corporation, Chengdu, China
[4] The Chinese University of Hong Kong [5] SIAT, Chinese Academy of Sciences, Shenzhen, China
[6] Guangdong Provincial Key Laboratory of Novel Security Intelligence Technologies
yang.sun@stu.hit.edu.cn, 23S051030@stu.hit.edu.cn, 21s051020@stu.hit.edu.cn
jianzhubao@gmail.com, bin.liang@cuhk.edu.hk, zxmm2@163.com
min.yang@siat.ac.cn, xuruifeng@hit.edu.cn

Abstract

End-to-end argumentation mining (AM) aims to extract the argumentation structure including argumentation components and their argumentation relations from text. Recent developments in end-to-end AM models have demonstrated significant progress by redefining the AM task as a sequence generation task, exhibiting simplicity and competitive performance. Nevertheless, these models overlook the integration of supplementary discourse structure information, a crucial factor for comprehending argumentation structures, resulting in suboptimal outcomes. In this study, we propose the DENIM framework, which generates discourse structure-aware prefixes for each layer of the generation model. These prefixes imbue the generation-based AM model with discourse structures, thereby augmenting the overall generation process. Moreover, we introduce a multi-task prompt coupled with a three-step decoding strategy, aiming to optimize the efficiency and effectiveness of argumentation structure decoding. Extensive experiments and analyses on two benchmark datasets show that DENIM achieves state-of-the-art performances on two AM benchmarks.

1 Introduction

Argumentation mining (AM) has recently received much research attention (Palau and Moens, 2009; Stede et al., 2019; Lawrence and Reed, 2020), which aims to analyze and understand argumentation texts to obtain argumentation structure.

Generally, AM generally comprises three subtasks following (Morio et al., 2022): 1) argumentation component segmentation (ACS) detects the boundaries of token-level argumentative segments, which are known as ACs; 2) argumentation component type classification (ACTC) classifies the segment-level ACs into the categories (i.e., Claim and Premise); 3) argumentation relation classification (ARC) further classifies the AR types (i.e., No-Relation, Support and Attack) between AC pair. The end-to-end AM task is highly challenging due to the complexity of simultaneously addressing these three subtasks within a unified framework.

Early studies focus on only a subset of the three subtasks (Niculae et al., 2017; Bao et al., 2021a) in AM. Recently, some studies have formulated the end-to-end AM as a dependency parsing task (Eger et al., 2017; Ye and Teufel, 2021). However, dependency parsing requires a complex pre-processing and post-processing process to match the argumentation structure with the elaborately designed dependency graph. Morio et al. (2022) tackled AM by combining sequence labeling and multi-class classification tasks within a multi-task learning framework. Inspired by the success of the generative methods, a generation-based end-to-end AM model (Bao et al., 2022) is proposed to transform the AM as a unified generation task.

Compared to traditional classification-based methods (Ye and Teufel, 2021; Morio et al., 2022), recently proposed generative AM models (Bao et al., 2022) have demonstrated simplicity in task decoding and competitive performance. However, existing generative approaches primarily focus on problem reformulation and, unlike traditional classification-based methods, do not incorporate additional discourse structure information, which has been proven effective for argument mining in prior works (Accuosto and Saggion, 2019, 2020). As shown in Figure 1, the discourse structure graph derived from the input text summarizes the semantic structure of the text, where the elementary discourse units (EDUs) are nodes and the discourse relations (e.g., Contrast) between EDUs form edges. In this graph, many nodes and edges

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

Findings of the Association for Computational Linguistics: ACL 2024, pages 11597–11613

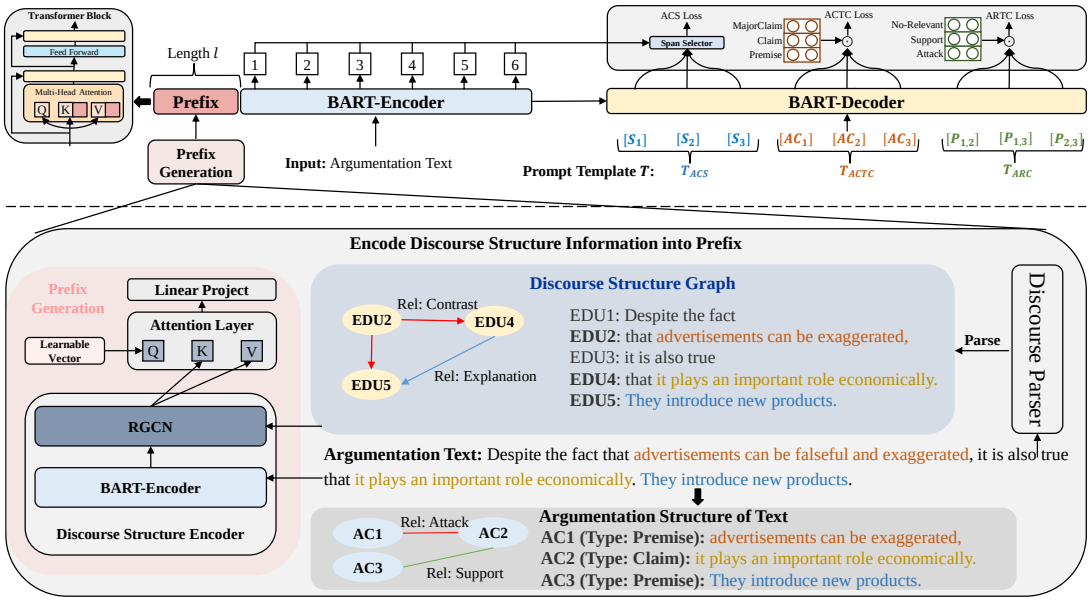


Figure 1: The architecture of DENIM, where the discourse structure graph of argumentation text in the blue block has a strong similarity to the argumentation structure in the grey block of the text and thus provides important clues for AM. Noted that the two BART-Encoders are identical.

share strong similarities with the argumentation structure. For example, the AC3 " They introduce new products" can be mapped to the EDU5, and its relative AC can be found using edge "Contrast". Hence, we argue that the discourse structure graph could provide important clues for generative-based AM models to figure out the argumentation structure, potentially enhancing overall performance and improving generalizability.

To address the aforementioned issue, we propose DENIM (Discourse structure-aware prEfix geNeration-based argumentatIon Mining), a generation-based end-to-end AM model that incorporates discourse structure information into prefix (Li and Liang, 2021) to guide the generation-based AM models. Specifically, we employ an additional discourse structure encoder to convert the input discourse structure graph into dense vectors. Then, these vectors will be disassembled and allocated to each Transformer layer within generation-based AM models as prefixes. These generated prefixes are converted into supplementary key and value matrices, exerting influence on the attention calculation process and guiding the generation mechanism.

Furthermore, we design a multi-task prompt using a fixed template and a three-step decoding strategy tailored for AM. DENIM initially produces spans of all ACs for ACS, subsequently classifies their types for ACTC, and ultimately identifies the relations between AC pairs for ARC. The computation process in each step adheres to a fixed format and is parallelized on GPUs. Compared to the previous generation-based AM model (Bao et al., 2022) using an autoregressive decoding strategy, our method demonstrates enhanced efficiency through task-specific decoding and improved controllability with a fixed prompt template.

Our contributions can be summarized as follows. (1) We propose a discourse structure-aware prefix to encode discourse structure information for the generation-based end-to-end AM model. (2) We design a multi-task prompt with a three-step decoding strategy for AM. This approach is more effective and efficient than the traditional autoregressive decoding strategy. (3) We conduct extensive experiments on two AM benchmark datasets. The experimental results present that our method significantly outperforms the strong baselines.

2 Methodology

DENIM uses BART (Lewis et al., 2019) as the base model, and augments it with discourse structure-aware prefix and multi-task prompt with a three-step decoding strategy, as depicted in Figure 1. To generate the discourse structure-aware prefix, we first employ a pre-trained discourse parser to extract the discourse structure graph of the input text (Section 2.2). Then, the graph is transformed into dense vectors through a discourse structure encoder.

Subsequently, these dense vectors will be disassembled and distributed to each layer of the BART so the generation is guided by the discourse structure information (Section 2.3). Lastly, we introduce a multi-task prompt with a three-step decoding strategy to alleviate the time-consuming and instability of sequence decoding for effectively and efficiently generating the argumentation structure (Section 2.4).

2.1 Task Definition

Following previous works (Morio et al., 2022), for the end-to-end AM, the input is a piece of argumentation text $W = \{w_1, w_2, \ldots, w_n\}$ consisting of n tokens. The first goal is to extract a set of ACs $A = \{a_i | a_i = (s_i, e_i)\}_i^m$ for ACS, where a_i is the i-th AC, s_i and e_i denote its start and end indexes respectively, and m denotes the number of ACs in the text. Next, the type c_i (i.e., *Claim* and *Premise*) of each extracted AC should be predicted for ACTC. Finally, the argumentation relation $rel_{(i,j)}$ (i.e., *No-relevant*, *Support* and *Attack*) between each AC pair (a_i, a_j) should be identified for ARC.

2.2 Discourse Structure Parsing

The first step of our method is to prepare the discourse structure graph (DSG) of the input text. We consider an external discourse parser called DMRST (Liu et al., 2021) to automatically parse the input text. As illustrated by Figure 1, the discourse parser encodes the input text into a DSG $G =< V, E >$, where each node $v_i \in V$ represents an EDU. The edge set, $E \subseteq V \times R \times V$, comprises edges $e_{i,j} \in \mathbb{R}^{|R|} \subseteq E$, each indicating a distribution over all discourse relations between two EDUs (v_i, v_j). Here, we consider three types of common discourse relation types $R = \{Expansion, Contingency, Comparison\}$ like (Pu et al., 2023) [1].

Note that we utilize the distributions over all discourse relation labels between two EDUs to enhance the heterogeneous graph, rather than restricting to the 1-best result (i.e., the relation with the highest predicted probability) inspired by (Pu and Sima'an, 2022). This strategy offers two advantages. Firstly, it mitigates error propagation from the external discourse parser by using relation distributions. Secondly, as suggested by Yung et al. (2022), multiple intrinsic relations can coexist simultaneously between EDUs. Representing these relations as a distribution rather than the 1-best result provides a more nuanced and accurate discourse structure. Thus, we argue that the logit output from the parser is more informative. It provides not just the n-best results but also captures the remaining uncertainty of predictions.

We validate the effectiveness of this strategy and our model's sensitivity to discourse structure information through experiments detailed in Appendix A.5.

2.3 Discourse Structure-Aware Prefix Generation

Next, to enable the discourse structure information to guide the generation-based AM model, we use the prefix (Li and Liang, 2021) as the bridge between them and inject the discourse structure information into the prefix. We employ a discourse structure encoder comprising a BART-Encoder and a Relational Graph Convolutional Network (RGCN) (Schlichtkrull et al., 2018) to model the discourse structure graph. The BART-Encoder first learns the initialized node representation and the RGCN models the graph structure by information propagation. Specifically, we utilize the BART-Encoder to encode the argumentative text W and obtain the hidden states $H^W = BART_{Encoder}(W)$. Then, for i-th node in the DSG $\mathcal{G}$, we mean-pool its context corresponding representations from H^W to obtain the initialized node feature h_i.

To let the nodes in DSG interact with each other and model the discourse structure information, we perform graph-based information propagation (Gilmer et al., 2017) to update node representations. Since DSG is a heterogeneous graph containing multiple types of edges, we utilize the relation-specific node-to-node information propagation mechanism in RGCN to model the DSG. Specifically, given a node $v \in \mathcal{G}$ at the c-th RGCN layer, the information propagation and aggregation operation is defined as follows:

$$\mathbf{h}_v^{c+1} = ReLU(\mathbf{h}_v^c + \sum_{r \in R} \sum_{u \in N_r(v)} \hat{E}_{v,r,u} W_r^c \mathbf{h}_u^c + b_r^c) \quad (1)$$

where $N_r(v)$ denotes the neighbors for node v connected with the edge of type r, ReLU is the ReLU activation function, W_r^c and b_r^c are the trainable parameters. $\hat{E}_{v,r,u} = \frac{E_{v,r,u}}{\sum_o E_{v,r,o}+1}$ is the normalized edge weight between the node v and u with relation r. Finally, we select the representation of

[1] Further details are presented in Appendix A.7

all nodes in the last layer as the updated node representations, which aggregate discourse structure information from a certain heterogeneous graph.

After that, we introduce an attention layer (Vaswani et al., 2017) and z learnable vectors as queries, where z is a hyperparameter determining the length of the used prefixes. These queries interact with the updated node representations, serving as both keys and values in the attention mechanism. The output of the attention layer is a set of dense vectors, denoted as P, which effectively condense the discourse structure information from the node representations.

We then transform these vectors into prefixes, similar to (Li and Liang, 2021), to be incorporated into our generation-based AM model. Concretely, this transformation involves partitioning P into L pieces, corresponding to the number of layers in the generation-based AM model, i.e., $P = \{P^1, \ldots, P^L\}$. For the i-th layer, the prefix is further divided into two matrices representing additional keys and values matrices: $P^i = \{K^i, V^i\}$, where K^i and V^i are the addition key and value matrices, and they can be further written as $K^i = \{k_1^i, \ldots, k_z^i\}$ and $V^i = \{v_1^i, \ldots, v_z^i\}$. k_*^i and v_*^i are vectors with the same hidden dimension in the i-th Transformer layer. These additional key and value matrices will be concatenated with the original key and value matrices in the attention block. Consequently, when calculating dot-product attention, the query at each position will be influenced by these discourse structure-aware prefixes.

It is worth noting that we assign different key-value pairs for different layers, offering two benefits. 1) the layer-wise queries and keys can exert strong control. 2) Different layers may need to embed varied information (Clark et al., 2019; Rogers et al., 2021). In addition, DENIM generates a distinct set of prefixes when the input text varies. The variation reflects the different discourse structure graph's presentation. This differs from the prefix tuning technique (Li and Liang, 2021) uses a fixed set of prefixes for all input instances.

We can integrate prefixes into the encoder self-attention blocks, decoder cross-attention blocks, or decoder self-attention blocks of our generation-based AM model. Based on our preliminary experiments in Appendix A.3, we observe that using prefixes in the self-attention blocks of the encoder and decoder works best in DENIM.

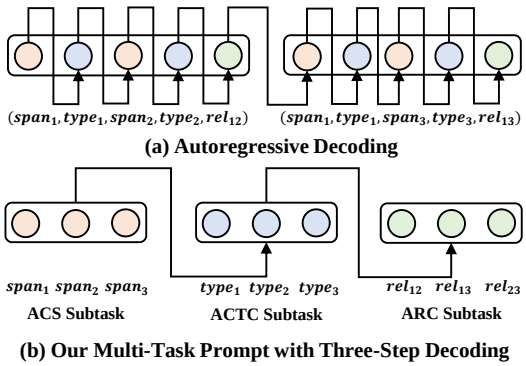


Figure 2: Comparison of decoding process between autoregressive decoding and our three-step decoding strategy. Top panel: the autoregressive decoding decodes the target sequence by token-by-token generation. Bottom panel: our three-step decoding employs a fixed prompt template and task-by-task decoding for AM.

2.4 Multi-Task Prompt with Three-Step Decoding

We design a more effective and efficient multi-task prompt with a three-step decoding strategy to extract the argumentation structure for AM, compared with the previous generation-based AM model (Bao et al., 2022) using an autoregressive decoding strategy. Specifically, as shown in the top panel of Figure 2, the previous generation-based AM model formulates AM as a sequence generation task and the target sequence comprises several tuples. Each tuple represents an AC pair and their relation in the format $[span_1, type_1, span_2, type_2, rel_{1,2}]$ where $span_i$ denotes the start/end indexes of i-th AC, $type_i$ is type (i.e., Claim and Premise) of i-th AC and $rel_{i,j}$ denote the argumentation relation (i.e., Support and Attack) between the AC pair (i, j). They apply an autoregressive decoding strategy to decode the target sequence by token-by-token generation. It not only is time-consuming but also may cause invalid predictions since the sequence generation is not fully controllable (Bao et al., 2022).

To mitigate the issues of time-consuming and uncontrollable token-by-token generation of autoregressive decoding (Bao et al., 2022), as discussed in Section 1, we introduce a multi-task prompt with a three-step decoding for AM, as shown in Figure 2. In particular, given an argumentative text W, we assume it has at most m ACs [2] and $m \times (m-1)/2$

[2]We heuristically set m to the maximum number of ACs in the train set.

AC pairs, the prompt template can be defined as:

$$
\begin{aligned}
T &= Concat(T_{ACS}, T_{ACTC}, T_{ARC}) \\
T_{ACS} &= [S_1] \dots [S_m] \\
T_{ACTC} &= [AC_1] \dots [AC_m] \\
T_{ARC} &= [P_{(1,2)}] \dots [P_{(m-1,m)}]
\end{aligned}
$$

Where $[S_i]$, $[AC_i]$ and $[P_{(i,j)}]$ placeholders [3] denote the span, type of AC a_i and relation between the AC pair (a_i, a_j) for ACS, ACTC and ARC subtasks, respectively. We use a fixed number of template tokens for all instances though different instances have different numbers of ACs. These redundant placeholders for each instance can be recognized as invalid tokens during training and inference. We feed the prompt template into the BART-Decoder to decode the argumentation structure for AM.

Training The training objective of DENIM is to predict gold labels based on the placeholders in the prompt template for AM. Specifically, DENIM is expected to derive the i-th AC span (s_i, e_i) from the input text W using the placeholder $[S_i]$ for the ACS subtask. For that, we employ the output hidden state h_i^S of the $[S_i]$ to get the span selector $\theta_i = \{\psi_{s_i}, \psi_{e_i}\}$, where $\psi_{s_i} = w_{start} h_i^S$, $\psi_{e_i} = w_{end} h_i^S$, and w_{start} and w_{end} are learnable parameters. Then, the start/end indexes (s_i, e_i) of the i-th AC can be obtained by calculating the distribution of each token in the input context W:

$$
\begin{aligned}
s_i &= \underset{0 \le k \le n}{\operatorname{argmax}} \operatorname{softmax}(\psi_{s_i} H^W) \\
e_i &= \underset{0 \le k \le n}{\operatorname{argmax}} \operatorname{softmax}(\psi_{e_i} H^W)
\end{aligned} \tag{2}
$$

For ACTC, DENIM predicts the type of the i-th AC using the output hidden state of the placeholder $[AC_i]$. To align the prediction of the placeholder $[AC_i]$ with the i-th AC span, we add the context representation $h_{a_i} = \frac{1}{e_i - s_i + 1} \sum_{k=s_i}^{e_i} H_k^W$ of the gold AC a_i into the embedding of the $[AC_i]$ before feeding it into the BART-Decoder. Similarly, we add the sum of context representations $h_{a_i,a_j} = \frac{1}{2}(h_{a_i} + h_{a_j})$ of the gold AC pair (a_i, a_j) and the embedding of their gold type into the embedding of the placeholder $[P_{(i,j)}]$ and feed it into the BART-Decoder. DENIM predicts the relation between AC pair (a_i, a_j) using the output hidden states of the placeholder $[P_{(i,j)}]$. The loss functions of all subtasks are cross-entropy losses.

[3]The placeholders are also called virtual words and implemented by using different specific tokens similar to eos token $< s >$ in the BART vocabulary.

Take Figure 1 as an example, DENIM is expected to enable $[S_1]$ to predict the AC span ("advertisements can be exaggerated"), then use $[AC_1]$ to predict the type ("Premise") of the AC a_1, and use $[P_{1,3}]$ to predict the relation ("Attack") between AC pair (a_1, a_3). If there is no predicted AC for one placeholder $[S_i]$, the model should predict the last position of the input W (i.e., specific token "</s>") and the $[AC_i]$ and $[P_{i,j}]$ are invalid.

Inference During inference, our prompt template utilizes a three-step decoding strategy, differing from the token-by-token autoregressive decoding in the prior generation model (Lewis et al., 2019; Bao et al., 2022). Initially, DENIM inputs all placeholders $[S_*]$ into the decoder to derive all AC spans. This step sets the foundation for subsequent decoding. In the second step, placeholders $[AC_*]$ with the context representation of the predicted AC spans are fed into the decoder to predict the type of ACs, as in the training phase. Finally, DENIM calculates the relation between all AC pairs using the placeholder $[P_{(*)}]$ with the sum of the context representation of the predicted AC pair and the embedding of the AC type. Notably, the decoding of the placeholders in the later subtask depends only on the result of placeholders in the former subtask. Thus, our prompt template only needs to be fed into the decoder three times, which saves time compared to autoregressive decoding. Furthermore, the prompt template serves as a strong control signal and defines the expected output format to ensure the output is valid.

Efficiency Considerations DENIM comprises three major components including a BART, an RGCN and a prompt template. The RGCN is required to model the discourse structure graph, with a time complexity of $O(|R||V|^2)$ that may cause efficiency considerations. In practice, this issue is minor for our experiments on the two datasets (AAEC and Abstract) and real scenarios, as the number [4] of EDUs in argumentation texts is consistently small in scale. Additionally, when incorporating the prefix into the BART-Encoder, the input context needs to be processed twice by the BART-Encoder. The first time is to construct the prefix and the second is to model the context representation of the input for the decoder. Its time cost is acceptable, as demonstrated in Appendix A.6.

[4]The average number of EDUs for the samples in the two datasets is 6 and 17, respectively.

3 Experimental Setup

Datasets To evaluate the effectiveness of our DENIM model, we conduct extensive experiments on two widely used AM datasets, named: AAEC (Stab and Gurevych, 2017) and AbstRCT (Mayer et al., 2020).

Evaluation Metrics We employ the same evaluation metrics as the previous works (Morio et al., 2022; Cheng et al., 2022; Guo et al., 2023), including F_1 score and macro averaged score (denoted as Macro). We calculate F_1 scores for determining the AC span for ACS. We adopt the F_1 and macro averaged score for ACTC and for determining the relation type (not including No-Relevant) for ARC. We also introduce the Link score (Kuribayashi et al., 2019), used to measure F_1 scores for identifying the existence of relations regardless of their types.

Baselines We compare DENIM with the following strong baseline models. For the AAEC dataset, we compare our model with five strong baselines, including **BiPAM** (Ye and Teufel, 2021), **BiPAM-syn** (Ye and Teufel, 2021), **BART-B** (Yan et al., 2021), **GMAM** (Bao et al., 2022) and **ST** (Morio et al., 2022). For the AbstRCT dataset, we compare our model with fhree strong baselines, which are **BART-B** (Yan et al., 2021), **GMAM** (Bao et al., 2022) and **ST** (Morio et al., 2022).

Implementation Details DENIM is implemented in PyTorch on an NVIDIA TESLA A100-PCIE-40GB and employs the BART base[5]. Our model is optimized using AdaW (Loshchilov and Hutter, 2017) with the learning rates of 3e-5 and weight decay of 1e-5 on both AbstRCT and AAEC datasets. For both datasets, we set the batch size to 4 and adopt dropout (Srivastava et al., 2014) with a dropout rate of 0.1 to avoid overfitting. We set the layer number L of RGCN to 1 because of its best performance. All experiments are performed five times with different random seeds, and the evaluation scores are averaged. Our code is available at `https://github.com/syiswell/DENIM`.

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 AAEC and AbstRCT datasets. For example, on AAEC, DENIM exceeds the current state-of-the-art (SOTA) method ST and obtains about 6.74% higher Macro score on the ACTC subtask. On AbstRCT, DENIM outperforms ST by 7.23% in terms of F_1 score on the ACS subtask. The experimental results verify the superiority of our method for AM.

We also observe that the generation-based models (i.e., BART-B and GMAM) surpass the dependency parsing-based methods (i.e., BiPAM and BiPAM-syn), indicating that formulating the end-to-end AM task as a generation task might be more effective than as a dependency parsing task. While BART-B and GMAM underperform the SOTA method ST, they typically offer a simpler decoding paradigm for AM. Based on the generation-based method, our DENIM performs better than all strong baselines by integrating a discourse structure-aware prefix and a multi-task prompt.

4.2 Ablation Study

To analyze the impact of different components in our proposed DENIM method, we conduct ablation studies in terms of removing RGCN (w/o RGCN), removing discourse structure-aware prefix (w/o Prefix), and removing multi-task prompt (w/o Prompt), respectively. Note that w/o Prompt means removing the prompt and transforming the task-specific decoding into token-by-token decoding while the target sequence is similar to the prompt format. The results are reported in Table 2.

We can observe that w/o RGCN degrades the performance, verifying that the discourse relations are beneficial for AM. w/o Prefix leads to further performance drops, demonstrating that discourse structure-aware prefixes can provide crucial clues to figure out the argumentation structure. Note that w/o Prefix outperforms previous generation-based methods (GMAM and BART-B) and SOTA method ST, verifying that our multi-task prompt with three-step decoding is more effective than them. Removing the multi-task prompt (w/o Prompt) leads to a noticeable drop in performance due to the absence of an efficient output format and strong control signals from the prompt.

4.3 Adaptability Experiment

Inspired by the recent success of Large Language Models (LLMs) (Min et al., 2023), we conduct additional experiments on two prominent LLMs, namely ChatGPT-3.5-Turbo and LLAMA2 (Tou-

[5]We implement BART using huggingface toolkit: https://huggingface.co/

Data	Model	ACS	ACTC		Link	ARC		AVG
			F1	Macro		F1	Macro	
AAEC	**BiPAM**	-	72.90	-	-	45.90	-	-
	BiPAM-syn	-	73.50	-	-	46.40	-	-
	BART-B	81.71	73.61	70.10	49.75	47.93	34.73	59.57
	GMAM	84.10	75.94	71.96	50.40	50.08	36.22	61.45
	ST	85.02	75.43	**73.49**	55.75	55.19	44.11	64.83
	DENIM (our)	**85.75**	**76.50**	73.33	**59.55**	**58.51**	**44.14**	**66.30 (+1.47)**
AbstRCT	**BART-B**	60.37	55.88	43.96	32.07	30.47	18.41	40.19
	GMAM	72.67	65.35	49.36	34.88	33.64	27.21	46.85
	ST	70.29	64.16	45.04	39.35	38.38	31.91	48.19
	DENIM (our)	**77.52**	**70.19**	**51.71**	**41.79**	**40.46**	**34.70**	**52.73 (+4.54)**

Table 1: Performance comparison on the AAEC and AbstRCT. **AVG** indicates the average value across all metrics. Our improvements over baselines are statistically significant with $p < 0.05$.

Data	Model	ACS	ACTC	Link	ARC	AVG
AAEC	DENIM	**85.75**	**74.92**	**59.55**	**51.33**	**66.30**
	w/o RGCN	84.39	73.11	57.79	49.80	64.67
	w/o Prefix	84.52	72.69	56.44	48.77	63.98
	w/o Prompt	84.68	68.86	47.33	38.83	57.90
AbstRCT	DENIM	**77.52**	**60.95**	**41.79**	**37.58**	**52.73**
	w/o RGCN	76.76	60.19	40.29	37.04	51.92
	w/o Prefix	76.57	60.33	38.90	34.40	50.82
	w/o Prompt	76.02	55.24	27.19	17.08	41.31

Table 2: The results of ablation study where we present the average value of F1 and Macro for ACTC and ARC.

vron et al., 2023). We employ the natural language prompt approach like (Madaan et al., 2022; Li et al., 2023) and few-shot in-context learning (here we employ 3-shot due to the limitation of input length) in ChatGPT-3.5-Turbo and LLAMA2 for AM. Figure 4 displays the format of the natural language prompt for AM. To evaluate the versatility of our method, we adapt it to fine-tune LLAMA2 using LORA (Hu et al., 2021), denoted by LLAMA2-DENIM. Furthermore, we fine-tune LLAMA2 with the natural language prompt approach (denoted by LLAMA2-FT) and LLAMA2-DENIM without prefix (w/o Prefix) as baselines. Each fine-tuning method uses a batch size of 1. Unfortunately, due to computational resource limitations, we are unable to perform fine-tuning experiments using LLAMA2 on the AbstRCT dataset. The results are presented in Table 3.

We observe that few-shot-based methods (i.e., LLAMA2 and ChatGPT-3.5-Turbo) significantly underperform fine-tuning methods (i.e., DENIM, LLAMA2-DENIM and LLAMA2-FT) on AAEC. Among these fine-tuning approaches, our DENIM framework with LLAMA2 (i.e., LLAMA2-DENIM) outperforms the w/o Prefix as well as the LLAMA-FT which is fine-tuned using natural language prompt, validating the effectiveness of our methods. In addition, LLAMA-DENIM performs worse than DENIM using BART as the base model. This discrepancy can be attributed to LLAMA-DENIM's large number of parameters (including trainable and non-trainable parameters) while the AM datasets have a small number of samples (see Data Statistics), which leads to severe overfitting.

4.4 Impact of Different Ways for Discourse Information Incorporation

We compare different ways to incorporate discourse structure information into the generation-based AM model: 1) discourse structure-aware prompts for Encoder (denoted by DENIM-E). We append the output representations of RGCN to the input text embeddings as the prompt. 2) encoding concatenation (denoted by DENIM-C). We concatenate the output representations of RGCN with the output representation of the BART-Encoder and feed them together to the BART-Decoder. 3) discourse structure-aware prompts for Decoder (denoted by DENIM-D). We concatenate the output representations of RGCN to the embeddings of the multi-task prompt template as part of the prompts. We present the results in Table 4. We can observe that DENIM outperforms all variants. An interesting finding is that DENIM-E is worse than the model w/o Prefix in Table 2. This is because the heterogeneous nature between discourse structure graphs and natural language sentences of encoder input would be confusing for models. The DENIM-C achieves better performance than DENIM-D and DENIM-E by concatenating the discourse structure information with high-level output representations of the encoder instead of natural language sentences and formatted prompts.

Data	Type	Model	ACS	ACTC		Link	ARC		AVG
				F1	Macro		F1	Macro	
AAEC	FT	DENIM	85.75	76.50	73.33	59.55	58.51	44.14	66.30
		LLAMA2-FT	78.37	64.61	61.90	46.14	44.59	29.00	54.10
		LLAMA2-DENIM	85.53	76.74	74.19	50.50	48.95	41.70	62.94
		-w/o Prefix	84.85	74.54	73.08	45.62	44.85	36.50	59.91
	FS	ChatGPT-3.5-Tubor	59.71	42.85	37.30	25.61	22.53	14.17	33.70
		LLAMA2	22.41	15.92	9.13	3.91	3.91	2.09	9.56
AbstRCT	FT	DENIM	77.52	70.19	51.71	41.79	40.46	34.70	52.73
	FS	ChatGPT-3.5-Tubor	65.91	58.82	47.98	29.79	26.37	12.07	40.15
		LLAMA2	22.94	19.90	11.42	6.03	5.70	2.26	11.38

Table 3: Performance comparison of the AAEC and AbstRCT. AVG indicates the average value across all metrics. FT and FS represent fine-tuning and few-shot learning approaches, respectively.

Data	Model	ACS	ACTC	Link	ARC	AVG
AAEC	DENIM	**85.75**	**74.92**	**59.55**	**51.33**	**66.30**
	DENIM-E	84.16	72.24	56.31	48.95	63.81
	DENIM-C	84.69	73.35	57.65	50.51	65.01
	DENIM-D	85.25	74.04	56.70	49.50	64.84
AbstRCT	DENIM	**77.52**	**60.95**	**41.79**	**37.58**	**52.73**
	DENIM-E	76.01	59.55	39.02	34.43	50.50
	DENIM-C	77.08	57.93	40.61	36.59	51.12
	DENIM-D	76.76	51.05	38.96	59.72	51.05

Table 4: The impact of different ways for Discourse Information Incorporation.

Data	Model	ACS	ACTC	Link	ARC	AVG
AAEC	DENIM	**85.75**	**74.91**	**59.55**	**51.33**	**66.30**
	DENIM-I	84.55	73.83	58.67	50.68	65.37
	DENIM-R	84.68	73.83	59.19	50.47	65.41
	DENIM-T	85.52	75.23	59.03	50.57	66.02
AbstRCT	DENIM	**77.52**	**60.95**	**41.79**	**37.58**	**52.73**
	DENIM-I	76.53	59.66	40.46	36.93	51.69
	DENIM-R	76.30	59.47	40.46	36.00	51.28
	DENIM-T	76.92	60.76	41.45	36.35	52.04

Table 5: The impact of different prompt approaches.

4.5 Impact of Different Prompt Template

We explore two multi-task prompt variants: 1). DENIM-I uses identical placeholders for each subtask to assess placeholder effects. 2). DENIM-R swaps ACTC and ARC placeholders to examine the impact of task order bias from the left-to-right generation paradigm of the BART. In addition, we introduce a two-step decoding strategy for our prompt (denoted by DENIM-T), where the ARC does not depend on the results of ACTC. During decoding, DENIM-T first decodes all AC spans and then predicts the type of AC and the relation between AC pairs simultaneously. We describe the prompt formats and decoding process for the three variants in detail in Appendix A.4.

The results of all variants of prompt templates on AAEC and AbstRCT are shown in Table 5. We observe that our prompt template gets better performance than DENIN-I, demonstrating different prompt tokens may contain specific semantics for different ACs and AC pairs. We also focus on DENIM-R making performance degradation compared to DENIM, confirming the negative effect caused by the order biases between tasks, which has been explored in previous work (Bao et al., 2022). Unsurprisingly, DENIM-T's performance is slightly worse than DENIM's due to placeholders in ARC not being able to access the information of types from the placeholders in ACTC, and vice versa.

4.6 Case Study

We conduct a case study to explain the benefit of integrating discourse information into AM intuitively. We compare DENIM and w/o Prefix and demonstrate two exemplary cases in Figure 3 to show the difference in their results.

Example A illustrates a typical case in which the EDUs in the DSG provide hints for the model to accurately segment the boundaries of the AC span. Specifically, the argumentation text is divided into five EDUs, two of which have strong similarities with the ACs of the argumentation structure. Without the DSG information, w/o Prefix will segment AC insufficiently, while DENIM can correctly segment two ACs in a lengthy sentence.

Example B shows how DSG helps DENIM perform ARC. In the DSG, both EDU2 and EDU3 have an "Elaboration" relationship pointing to EDU1. Given the high similarity of EDU1, EDU2 and EDU3 to AC1, AC2 and AC3, respectively, DENIM accurately identifies two AC pairs (AC1, AC2) and (AC1, AC3) with a "Support" relation, while w/o Prefix struggles to capture the relationship between AC pairs.

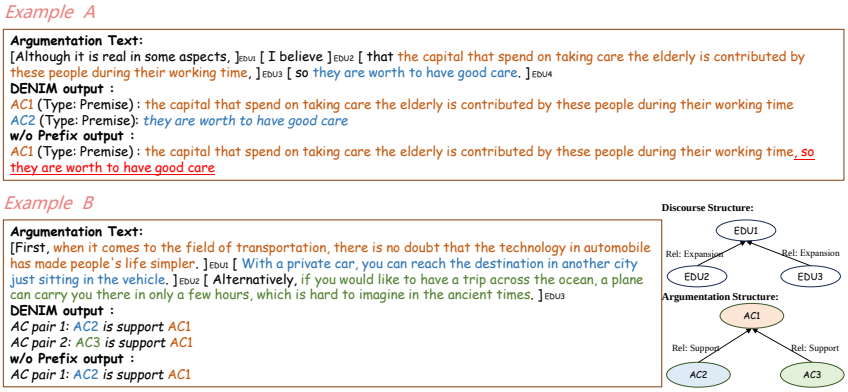


Figure 3: Two examples of how discourse structure information helps the generation of argumentation structure. A text fragment in a bracket is an EDU and a fragment of the same color (except black) is an AC.

5 Related Work

Argumentation Mining Argumentation mining aims to extract the argumentation structure from the argumentation text, which involves three subtasks (Lawrence and Reed, 2020; Bao et al., 2021b; Cheng et al., 2021; Sun et al., 2022; Guo et al., 2023; Chen et al., 2023). Early work (Persing and Ng, 2016) performed joint inference in an Integer Linear Programming (ILP) framework. With the success of deep learning, Eger et al. (2017) investigated neural techniques for AM and formalized the end-to-end AM task into multiple other tasks. Ye and Teufel (2021) extended the approach of (Eger et al., 2017) using biaffine dependency parsing and achieved promising performance. However, it requires tedious pre- and postprocessing. Bao et al. (2022) proposed a generative framework with a constrained pointer mechanism and a reconstructed positional encoding for end-to-end AM. Morio et al. (2022) proposed a Longformer (Beltagy et al., 2020) based model with biaffine functions for AM.

In addition, some works explore incorporating discourse information for AM. (Green, 2010) introduced a non-deep learning approach that leverages Rhetorical Structure Theory to analyze argument structures within biomedical corpora. (Stede et al., 2016) introduced two theories, including Rhetorical Structure Theory and Segmented Discourse Representation Theory, to annotate and analyze argumentative texts. (Accuosto and Saggion, 2019) proposed an LSTM-based transfer learning approach that uses contextual representations learned from discourse parsing tasks as input for argument mining models. (Accuosto and Saggion, 2020) employed the annotated discourse units and relations as auxiliary information and proposed two transfer learning approaches (i.e., multi-task learning and sequential learning) with BiLSTM and CRF models for AM. (Chistova, 2023) proposed a deep dependency parsing model to explore the relationship between rhetorical and argument structures. Different from them, we explore how to incorporate this additional information into a generation-based end-to-end AM and enhance it with prefix and prompt learning.

Prefix tuning and prompt tuning Prompt tuning (Liu et al., 2023) has achieved desirable performance in the field of natural language processing (NLP) (Schick and Schütze, 2021; Wang et al., 2022; Li and Liang, 2021; Ma et al., 2022; Hsu et al., 2023). Prefix tuning (Li and Liang, 2021), a member of the prompt-based tuning family, can trigger the desired generation of PLMs by only optimizing small continuous prefix vectors. To the best of our knowledge, we are the first to explore the potential of prefix tuning with discourse information for AM and propose an effective and efficient multi-task prompt with a three-step decoding strategy.

6 Conclusion

In this paper, we designed discourse structure-aware prefixes, which introduce discourse structure information to the generation-based AM model, thereby enhancing the generation. Additionally, we introduce a multi-task prompt complemented by a three-step decoding strategy, optimizing the efficiency and effectiveness of argumentation structure decoding. Extensive experiments and analyses on two benchmarks show that our method outperformed strong baselines significantly.

Limitation

To point out future research direction for generation-based AM models, we performed error analysis on 100 cases where our DENIM made mistakes for ACS, ACTC and ARC subtasks. On the ACS subtask, we identified two common types of errors: (1) incorrect AC span due to short segment. DENIM has lower confidence in the segmentation of short ACs compared to longer ACs. In particular, if an AC is a short segment, it tends not to be predicted as an AC or will be incorrectly spliced with subsequent clauses as an AC. (2) error AC combined with a main clause. For example, "It is generally understood that", "Many studies have pointed out that", etc. The former is not part of AC, while the latter is part of AC. DENIM finds it difficult to distinguish these subtle semantic differences. In DSG, both such examples are segmented into EDUs, which cannot provide effective hints for DENIM at the discourse structure level. For ACTC, the most serious problem is the error extraction of AC span, resulting in incorrect AC classification. In addition, we discover that there is a type bias for different positions of ACs. For instance, DENIM tends to predict the first AC as "Claim" and the last AC as "Premise". This is because DENIM overfits the type distribution of ACs at different positions using the prompt placeholders. Maybe we can adopt a debias approach to alleviate this issue. On the ARC task, we find that DENIM captures the connection between long-distance ACs so excessively that it generates additional AC pairs. In the argumentation structure, AC pairs with relationships should not be too far apart, or it won't follow the argumentation structure habits of human texts. Therefore, we suggest that during ARC tasks, incorporating distance loss between ACs can guide the model to focus more on the connection between ACs with moderate distances, which could potentially address the issue.

Last, we discuss the prefix technique proposed, which utilizes discourse structure graphs generated by a pre-trained discourse parser. These graphs, while effective, are not infallible and may contain imperfections, leading to potential error propagation issues in DENIM. Although employing distributions over all discourse relation labels helps mitigate this problem, it is not a perfect solution. A prospective approach to address this is to treat AM and discourse parsing as concurrent tasks within a multi-task learning framework. We leave this integrated approach as our 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 Key Research and Development Program of China (2022YFF0902100), the National Natural Science Foundation of China (62176076), Natural Science Foundation of Guangdong (2023A1515012922), the Shenzhen Foundational Research Funding (JCYJ20220818102415032), the Major Key Project of PCL2021A06, Guangdong Provincial Key Laboratory of Novel Security Intelligence Technologies (2022B1212010005).

References

Pablo Accuosto and Horacio Saggion. 2019. Transferring knowledge from discourse to arguments: A case study with scientific abstracts. In *Proceedings of the 6th Workshop on Argument Mining*, pages 41–51.

Pablo Accuosto and Horacio Saggion. 2020. Mining arguments in scientific abstracts with discourse-level embeddings. *Data & Knowledge Engineering*, 129:101840.

Katherine Atwell, Anthony Sicilia, Seong Jae Hwang, and Malihe Alikhani. 2022. The change that matters in discourse parsing: Estimating the impact of domain shift on parser error. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 824–845.

Jianzhu Bao, Chuang Fan, Jipeng Wu, Yixue Dang, Jiachen Du, and Ruifeng Xu. 2021a. A neural transition-based model for argumentation mining. 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 6354–6364.

Jianzhu Bao, Yuhang He, Yang Sun, Bin Liang, Jiachen Du, Bing Qin, Min Yang, and Ruifeng Xu. 2022. A generative model for end-to-end argument mining with reconstructed positional encoding and constrained pointer mechanism. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 10437–10449.

Jianzhu Bao, Bin Liang, Jingyi Sun, Yice Zhang, Min Yang, and Ruifeng Xu. 2021b. 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.

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

Lynn Carlson and Daniel Marcu. 2001. Discourse tagging reference manual. *ISI Technical Report ISI-TR-545*, 54(2001):56.

Guizhen Chen, Liying Cheng, Luu Anh Tuan, and Lidong Bing. 2023. Exploring the potential of large language models in computational argumentation. *arXiv preprint arXiv:2311.09022*.

Liying Cheng, Lidong Bing, Ruidan He, Qian Yu, Yan Zhang, and Luo Si. 2022. Iam: A comprehensive and large-scale dataset for integrated argument mining tasks. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2277–2287.

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.

Elena Chistova. 2023. End-to-end argument mining over varying rhetorical structures. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 3376–3391.

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.

Steffen Eger, Johannes Daxenberger, and Iryna Gurevych. 2017. Neural end-to-end learning for computational argumentation mining. *arXiv preprint arXiv:1704.06104*.

Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. 2017. Neural message passing for quantum chemistry. In *International conference on machine learning*, pages 1263–1272. PMLR.

Nancy L Green. 2010. Representation of argumentation in text with rhetorical structure theory. *Argumentation*, 24:181–196.

Jia Guo, Liying Cheng, Wenxuan Zhang, Stanley Kok, Xin Li, and Lidong Bing. 2023. Aqe: Argument quadruplet extraction via a quad-tagging augmented generative approach. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 932–946.

I-Hung Hsu, Zhiyu Xie, Kuan-Hao Huang, Prem Natarajan, and Nanyun Peng. 2023. Ampere: Amr-aware prefix for generation-based event argument extraction model. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10976–10993.

Edward J Hu, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. 2021. Lora: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Tatsuki Kuribayashi, Hiroki Ouchi, Naoya Inoue, Paul Reisert, Toshinori Miyoshi, Jun Suzuki, and Kentaro Inui. 2019. An empirical study of span representations in argumentation structure parsing. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4691–4698.

John Lawrence and Chris Reed. 2020. Argument mining: A survey. *Computational Linguistics*, 45(4):765–818.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension.

Peng Li, Tianxiang Sun, Qiong Tang, Hang Yan, Yuanbin Wu, Xuanjing Huang, and Xipeng Qiu. 2023. CodeIE: Large code generation models are better few-shot information extractors. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Toronto, Canada. Association for Computational Linguistics.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2023. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Computing Surveys*, 55(9):1–35.

Zhengyuan Liu, Ke Shi, and Nancy Chen. 2021. DMRST: A joint framework for document-level multilingual RST discourse segmentation and parsing. In *Proceedings of the 2nd Workshop on Computational Approaches to Discourse*, pages 154–164, Punta Cana, Dominican Republic and Online. Association for Computational Linguistics.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Yubo Ma, Zehao Wang, Yixin Cao, Mukai Li, Meiqi Chen, Kun Wang, and Jing Shao. 2022. Prompt for extraction? paie: Prompting argument interaction for event argument extraction. *arXiv preprint arXiv:2202.12109*.

Aman Madaan, Shuyan Zhou, Uri Alon, Yiming Yang, and Graham Neubig. 2022. Language models of code are few-shot commonsense learners. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 1384–1403.

Tobias Mayer, Elena Cabrio, and Serena Villata. 2020. Transformer-based argument mining for healthcare applications. In *ECAI 2020*, pages 2108–2115. IOS Press.

Bonan Min, Hayley Ross, Elior Sulem, Amir Pouran Ben Veyseh, Thien Huu Nguyen, Oscar Sainz, Eneko Agirre, Ilana Heintz, and Dan Roth. 2023. Recent advances in natural language processing via large pre-trained language models: A survey. *ACM Computing Surveys*, 56(2):1–40.

Gaku Morio, Hiroaki Ozaki, Terufumi Morishita, and Kohsuke Yanai. 2022. End-to-end argument mining with cross-corpora multi-task learning. *Transactions of the Association for Computational Linguistics*, 10:639–658.

Vlad Niculae, Joonsuk Park, and Claire Cardie. 2017. Argument mining with structured svms and rnns. *arXiv preprint arXiv:1704.06869*.

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*, pages 98–107.

Isaac Persing and Vincent Ng. 2016. End-to-end argumentation mining in student essays. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1384–1394.

Dongqi Pu and Khalil Sima'an. 2022. Passing parser uncertainty to the transformer: Labeled dependency distributions for neural machine translation. In *Proceedings of the 23rd Annual Conference of the European Association for Machine Translation*, pages 41–50.

Dongqi Pu, Yifan Wang, and Vera Demberg. 2023. Incorporating distributions of discourse structure for long document abstractive summarization. *arXiv preprint arXiv:2305.16784*.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2021. A primer in bertology: What we know about how bert works. *Transactions of the Association for Computational Linguistics*, 8:842–866.

Timo Schick and Hinrich Schütze. 2021. Exploiting cloze questions for few shot text classification and natural language inference.

Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *The Semantic Web: 15th International Conference, ESWC 2018, Heraklion, Crete, Greece, June 3–7, 2018, Proceedings 15*, pages 593–607. Springer.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958.

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

Manfred Stede, Stergos Afantenos, Andreas Peldzsus, Nicholas Asher, and Jérémy Perret. 2016. Parallel discourse annotations on a corpus of short texts. In *10th International Conference on Language Resources and Evaluation (LREC 2016)*, pages 1051–1058.

Manfred Stede, Jodi Schneider, and Graeme Hirst. 2019. *Argumentation mining*. Springer.

Yang Sun, Bin Liang, Jianzhu Bao, Min Yang, and Ruifeng Xu. 2022. Probing structural knowledge from pre-trained language model for argumentation relation classification. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 3605–3615.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Zihan Wang, Peiyi Wang, Tianyu Liu, Binghuai Lin, Yunbo Cao, Zhifang Sui, and Houfeng Wang. 2022. Hpt: Hierarchy-aware prompt tuning for hierarchical text classification. *arXiv preprint arXiv:2204.13413*.

Hang Yan, Junqi Dai, Xipeng Qiu, Zheng Zhang, et al. 2021. A unified generative framework for aspect-based sentiment analysis. *arXiv preprint arXiv:2106.04300*.

Yuxiao Ye and Simone Teufel. 2021. End-to-end argument mining as biaffine dependency parsing. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 669–678.

Frances Yung, Kaveri Anuranjana, Merel Scholman, and Vera Demberg. 2022. Label distributions help implicit discourse relation classification. In *Proceedings of the 3rd Workshop on Computational Approaches to Discourse*, pages 48–53.

A Appendix

A.1 Data Statistics

- **AAEC** (Stab and Gurevych, 2017): The AAEC dataset comprises 420 essays with 1833 paragraphs. There are three types of

ACs (i.e., *MajorClaim*, *Claim* and *Premise*) and two types of ARs (i.e., *Support* and *Attack*). Each AC has at most one outgoing AR so the argumentation graph of the paragraph can be either directed trees or forests. We divide this dataset into a training set of 1464 ACs and a testing set of 369 ACs, and randomly choose 10% of the training set as the validation set, which is consistent with previous works (Morio et al., 2022).

- **AbstRCT** (Mayer et al., 2020): The AbstRCT includes abstracts of randomized controlled trials (RCTs) from the MEDLINE database. All ACs are classified into three types: *MajorClaim*, *Claim* and *Evidence*. The ARs have three types: *Support*, *Attack* and *Partial-attack*. Each AC may have several outgoing ARs, thus the argumentation graph is of non-tree structure. We used 350 training, 50 development, and 100 test texts following (Morio et al., 2022).

The detailed statistics of the AAEC and AbstRCT datasets are summarized in Table 6.

Type	AAEC	AbstRCT
Text	1833	500
Train	1464	350
Test	369	100
Components	6089	3279
Relations	3832	2060

Table 6: The statistics of the AAEC and AbstRCT datasets.

A.2 Baselines

We compare our proposed model with the following baselines:

- **BiPAM** (Ye and Teufel, 2021): This method applies BERT as the base model and transforms AM as a biaffine dependency parsing task for end-to-end AM.

- **BiPAM-syn** (Ye and Teufel, 2021): The BiPAM model is enhanced by explicit syntactic information produced by the Stanford syntactic dependency parser.

- **ST** (Morio et al., 2022): This method proposes an end-to-end AM model based on Longformer (Beltagy et al., 2020) and the biaffine function.

Data	Type	ACS	ACTC	Link	ARC	AVG
AAEC	TTT	84.44	72.75	59.62	51.63	65.47
	TTF	84.02	71.71	59.96	51.48	65.06
	TFT	85.75	74.81	59.55	51.33	**66.30**
	FTT	85.10	73.92	58.31	47.89	64.51
	TFF	85.21	73.70	60.05	49.89	65.41
	FTF	84.66	72.86	58.61	51.46	65.32
	FFT	83.78	72.38	59.06	49.96	64.59
	FFF	84.52	72.69	56.44	48.77	63.98
AbstRCT	TTT	76.98	61.02	40.23	37.62	52.41
	TTF	76.30	59.92	40.40	34.85	51.04
	TFT	77.52	60.95	41.79	37.58	**52.73**
	FTT	78.36	60.62	40.29	36.88	52.27
	TFF	78.03	60.40	40.79	37.96	52.59
	FTF	76.89	59.38	39.70	34.92	50.91
	FFT	76.16	60.83	41.30	35.39	51.65
	FFF	76.57	60.33	38.90	34.40	50.82

Table 7: The results of the additional ablation study where we present the average value of F1 and Macro for ACTC and ARC. The letter "T" in the Type denotes true and the letter "F" denotes False. The three letters in the Type denote whether or not to use prefixes in the encoder's self-attention blocks, the decoder's cross-attention blocks, and the decoder's self-attention blocks, respectively. e.g. TTT denotes the use of prefixes in all three blocks respectively, and FFF denotes the use of prefixes in none of the three blocks.

- **BART-B** (Yan et al., 2021): The model converts all aspect-based sentiment analysis subtasks into a unified generative formulation, which is adapted for extracting argument structure by (Bao et al., 2022).

- **GMAM** (Bao et al., 2022): A generative end-to-end AM model reformulates AM as a sequence generation task, enhanced by reconstructed positional encoding and constrained pointer mechanism. Noted the BART-B is the basic model of GMAM without augmentation mechanism.

A.3 Additional Ablation Study

Here, we explore which blocks (i.e., self-attention blocks in the encoder, self-attention blocks in the decoder, or cross-attention blocks in the decoder) we should place prefixes into. We find that (1) employing prefixes consistently improves the performance of our methods. (2) Using the prefixes only in the self-attention blocks of the encoder outperforms using it in another block. (3) We obtained the best performance by using prefixes in the self-attention blocks of the encoder and decoder, instead of using prefixes in all blocks. This may be because reusing prefixes in the self-attention and

Input:
The text is "Despite the fact that advertisements can be exaggerated, it is also true that it plays an important role economically. They introduce new products. ". The type of argumentation component and argumentation relation in the text:

Output:
"advertisements can be exaggerated." is Premise.
"it plays an important role economically." is Claim.
"They introduce new products." is Premise.
the argument relation between "advertisements can be exaggerated." and "it plays an important role economically." is "Attack".
the argument relation between "it plays an important role economically." and "They introduce new products." is "Support".

Figure 4: The format of natural language prompts for AM.

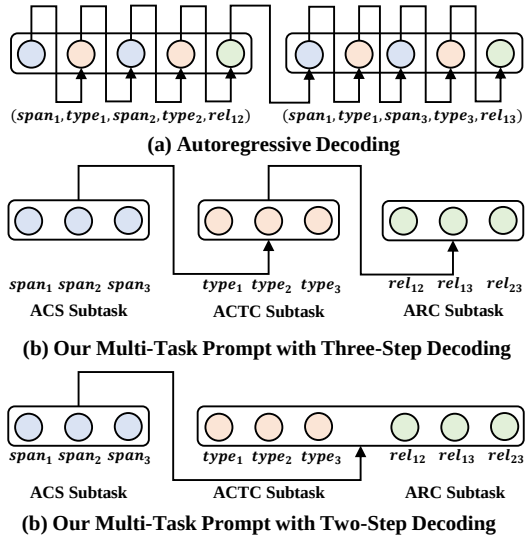


Figure 5: Comparison of the decoding process between autoregressive decoding and our three-step decoding strategy. Top panel: the autoregressive decoding decodes the target sequence by token-by-token generation. Middle panel: our three-step decoding employs a fixed prompt template and task-by-task decoding for AM. Bottom panel: our two-step decoding first decodes ACS and then ACTC and ARC simultaneously.

cross-attention blocks of the decoder introduces redundant information.

A.4 Impact of Different Prompt Template

We study two variants of our multi-task prompt: 1). prompt template with identical task placeholders (denoted by DENIM-I). To explore the effect of placeholders in the multi-task prompt, we employ the same placeholder for different ACs or AC pairs in each subtask. 2). prompt template with task order reverse (denoted by DENIM-R). To study the effect of task order biases in the left-to-right generation paradigm of the BART decoder for DENIM, we consider reversing the generation order between ACTC and ARC. For that, we reverse the position of placeholders for ACTC and the position of placeholders for ARI. The format of the two prompt templates can be viewed in Table 8.

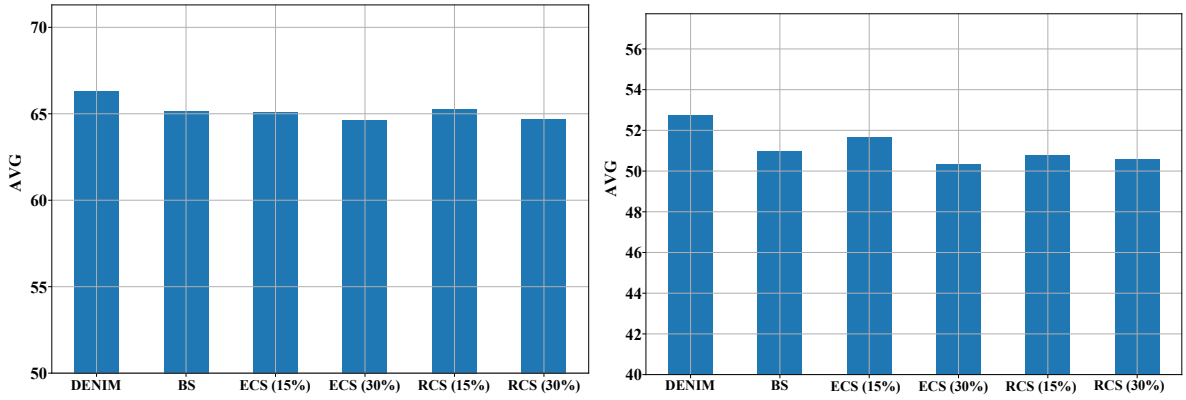
In addition, we introduce a two-step decoding strategy for our prompt (denoted by DENIM-T) as shown in the bottom panel of Figure 5, where the ARC does not depend on the results of ACTC compared with the three-step decoding. During decoding, DENIM first inputs all placeholders $[S_*]$ into the decoder to derive all AC spans, which is the same as the first step in three-step decoding. Then, placeholders $[AC_*]$ with the context representation of the predicted AC spans, and $[P_{(*)}]$ with the sum of the context representation of the two AC spans of the predicted AC pair, are fed into the decoder together. The placeholders $[AC_*]$ and $[P_{(*)}]$ are employed to predict the type of ACs and the relation between AC pairs simultaneously.

A.5 Discussion of Different Discourse Structure Graph Construction

We utilize the distributions over all discourse relation labels between two EDUs to strengthen the heterogeneous graph, rather than a limitation to the 1-best result (i.e., the relation with the highest predicted probability) inspired by (Pu and Sima'an, 2022). This approach offers two advantages. Firstly, it mitigates error propagation from the external discourse parser by using relation distributions. Since the parser is known to perform poorly on out-of-domain data (Atwell et al., 2022), and may hence propagate errors into the AM model. Secondly, as Yung et al. (2022) suggests, multiple intrinsic relations can exist simultaneously between EDUs. Representing these relations as a distribu-

Type	Templates
DENIM	$[D][S_1]\dots[S_m][AC_1]\dots[AC_m][P_{(1,2)}]\dots[P_{(m-1,m)}]$
DENIM-I	$[D][S]\dots[S][AC]\dots[AC][P]\dots[P]$
DENIM-R	$[D][S_1]\dots[S_m][P_{(1,2)}]\dots[P_{(m-1,m)}][AC_1]\dots[AC_m]$

Table 8: The format of different prompt templates.



(a) AAEC

(b) AbstRCT

Figure 6: The result of different discourse structure graph construction approaches on AAEC and AbstRCT, where the value in parentheses is k corresponding to each method.

Data	Model	Train (min)	Inference (sec)
AAEC	BART-B	0.12	119.60
	GMAM	0.13	129.52
	ST	1.62	17.40
	DENIM	0.80	21.86
	w/o Prefix	0.58	15.63
	w/o Prompt	0.65	51.24
AbstRCT	BART-B	0.16	151 29
	GMAM	0.16	117.13
	ST	0.79	5.32
	DENIM	0.79	7.72
	w/o Prefix	0.62	6.79
	w/o Prompt	0.76	37.37

Table 9: Comparison of the Training Time per epoch (minutes) and Inference Time in the test set (second) on AAEC and AbstRCT.

tion rather than the 1-best result provides a more nuanced and accurate discourse structure. Thus, we argue that the logit output from a discourse parser is more informative. It provides not just the n-best results but also captures the remaining uncertainty associated with these predictions.

To assess the efficacy of the proposed strategy, we conduct a set of experiments, incorporating a different variant. The variant termed the 1-best strategy (BS for short), utilizes the 1-best result to connect two nodes. The results are presented in Figure 6. Compared with the 1-best strategy, our method exhibits a consistent superiority across all metrics. This enhancement is attributed to our strategy's dual focus: mitigating error propagation from the discourse parser by leveraging a distribution over all relations, and acknowledging the potential coexistence of multiple intrinsic relations between EDUs to construct a more nuanced and accurate discourse structure.

Further exploration into the sensitivity of our method to discourse information involved the implementation of two additional variants. The second variant termed the EDU contamination strategy (ECS), introduces random contamination of $k\%$ of EDUs by changing its start/end indexes. The third variant, known as the relation contamination strategy (RCS), involves random contamination of $k\%$ of relations between two EDUs by substituting them with alternate relations. Surprisingly, we observed only a marginal decline in the model's performance with increasing k values, significantly outperforming the previous state-of-the-art (SOTA) standard. These findings underscore the pivotal role of discourse information for AM and demonstrate our method's commendable robustness in handling discourse information.

Relation	AAEC		AbstRCT	
	Overlap	Non-Overlap	Overlap	Non-Overlap
Attribution	3	179	2	21
Comparison	0	8	0	41
Condition	1	212	0	38
Enablement	10	446	2	201
Manner-Means	3	77	4	387
Same-Unit	4	169	3	177
Background	192	875	97	397
Cause	394	276	106	136
Contrast	322	508	384	285
Elaboration	1032	757	253	492
Evaluation	171	26	173	26
Explanation	317	124	3	24
Joint	730	1028	1255	1515
Summary	4	5	0	3
Temporal	34	40	13	104
TextualOrganization	0	0	0	0
Topic-Change	0	0	0	0

Table 10: Statistical analysis of the overlap between EDU and AC in various discourse relations. The frequency of the first six discourse relations appearing in non-AC text segments is significantly higher than in cases of overlaps with ACs.

Data	Method	ACS	ACTC		Link	ARC		AVG
			F1	Macro		F1	Macro	
AAEC	Unlabeled-DSG	84.76	74.26	71.32	56.92	56.39	38.53	63.70
	Unlabeled-Exc-DSG	85.63	75.01	71.15	56.98	56.33	40.46	64.26
	Full-DSG	85.44	74.38	72.01	57.76	56.27	41.62	64.58
	Exc-DSG	84.25	74.34	71.90	58.78	58.02	44.06	65.22
	Full-Random-DSG	84.30	74.07	71.39	57.61	56.30	42.70	64.40
	Exc-Random-DSG	84.46	74.29	70.37	58.55	57.24	43.65	64.76
	DENIM	**85.75**	**76.50**	**73.33**	**59.55**	**58.51**	**44.14**	**66.30**
AbstRCT	Unlabeled-DSG	76.27	70.00	50.66	39.24	37.67	33.47	51.22
	Unlabeled-Exc-DSG	76.76	69.71	50.67	40.29	39.47	34.60	51.92
	Full-DSG	77.16	69.90	51.27	40.07	37.80	31.89	51.35
	Exc-DSG	77.07	70.01	51.57	41.33	39.95	33.18	52.19
	Full-Random-DSG	77.29	69.74	50.36	38.91	37.38	34.02	51.28
	Exc-Random-DSG	77.20	70.04	50.68	40.65	38.92	32.28	51.63
	DENIM	**77.52**	**70.19**	**51.71**	**41.79**	**40.46**	**34.70**	**52.73**

Table 11: The impact of selection and categorization of discourse relations.

A.6 Computational Cost

We investigate the computational cost of baseline methods and our DENIM model in training and inference. For a fair comparison, all these models use the same batch size of 4 in training and 1 in inference. Table 9 shows the training time and inference time on the AAEC and AbstRCT.

DENIM is faster than or comparable to the SOTA model (i.e., ST) for training, but slightly slower than the base model for inference. For example, DENIM shows an average increase of 0.003s and 0.007s in inference time per sample in AAEC and AbstRCT, respectively, compared to ST, which is acceptable in practice. In addition, during inference, our DENIM is more efficient than previous generation-based baselines (i.e., BART-B and GMAM) by a factor of 5-20 owing to the multi-task prompt with a three-step decoding, although additional overhead is added to the training process.

Comparing DENIM with w/o Prefix, the prefix introduces a small amount of extra time (almost 13.2 seconds on AAEC and 10.2 seconds on AbstRCT for one epoch) in both the training and inference phases (6.23 seconds on PE and 0.93 seconds CDCP for all instances). This additional cost is acceptable.

A.7 Selection and Categorization of Discourse Relation

We notice that some discourse relations parsed by the DMRST parser may be irrelevant to the argument mining task, and removing these irrelevant

Group	Relation	Description	Group Description
1	Background	Elaborates on the objective background of the Nucleus.	These relations share a significant characteristic where one unit supplements or further elaborates on the information of another unit.
	Elaboration	Additional information and details of the Nucleus.	
	Evaluation	An evaluation or conclusion of the Nucleus.	
	Summary	Summarizes the Nucleus, without adding additional information.	
	Topic-Comment	Comments and explains the Nucleus (topic).	
2	Cause	Causes (subjective or objective reasoning).	These relations often denote that one unit results from or is the cause of another unit.
	Explanation	Explanation (both subjective and objective).	
3	Contrast	Contrast and transition, tending towards objective.	These relations often denote that one unit is of the same informational level as another unit, distinguishing them from the first set of relations.
	Joint	Lists and 'or' conditions.	
	Temporal	Sequences, such as 'when', 'before', etc.	
	Textual-Organization	Textual division and connection, without semantics.	
	Topic-Change	Topic change, semantic leaps.	

Figure 7: The definition and the description of the discourse relations.

relationships may benefit the AM task. A similar operation can be seen in prior work (Pu et al., 2023). For that, we analyze the number of overlaps between EDUs and ACs under each discourse relationship in the training set, as shown in Table 10. We observed that for relations such as Comparison, Condition, Enablement, and Manner-Means, the number of overlaps between EDUs with these relations and ACs was minimal, while the non-overlapping instances were significantly higher. We believe that EDUs that share text fragments with ACs provide more accurate cues for AM. To minimize the impact of these EDUs with low relevance and their associated relations, we chose to exclude these relations (named irrelevant relations) (The top six in Table 10).

To validate whether the irrelevant relations would affect performance, we explore four variants: 1) we employ all discourse relations to construct the DSG (denoted by Full-DSG). 2) we exclude irrelevant discourse relations and use the remaining relations to construct the DSG. (denoted by Exc-DSG). 3) we limit the difference among discourse relations and set the weight of edges in the DSG to 1 (denoted by unlabeled-DSG). 4) we set the weight of the edges with irrelevant relations to 0 in the DSG, while the weight of the remaining edges is set to 1 (denoted by unlabeled-Exc-DSG). The experimental results, presented in Table 11, indicate that excluding irrelevant discourse relations is beneficial. Our strategy suggests that future research could focus on more effectively eliminating noise in discourse relations and better aligning them with argumentative relations.

In addition, we refer to the definitions of (Carlson and Marcu, 2001) to obtain the semantics of each discourse relation, as shown in Figure 7. Then, inspired by (Pu et al., 2023), we manually group relations with similar semantics, such as Reason and Explanation, and present the grouping description in Figure 7. To validate the effectiveness of our grouping strategy, we conduct additional experiments, including applying the random grouping strategy to both Full-DSG and Exc-DSG variants (denoted by Full-Random-DSG and Exc-Random-DSG, respectively). The results of these comparisons, as shown in Table 11, demonstrate the rationale of our approach.

We recognize that the strategies we employed for the selection and grouping of discourse relations exhibit certain limitations. The specific grouping strategies may confuse the true features of categories, which merit further investigation in future research endeavors.