Argumentative Relationship Recognition Based on End-to-End Multitask Learning

Jingyuan Huang
National University of Defense Technology
jingyuanhuang@nudt.edu.cn

Quntian Fang
National University of Defense Technology
fangquntian@nudt.edu.cn

Sijie Wang
National University of Defense Technology
970831193@qq.com

Zhiliang Tian*
National University of Defense Technology
tianzhiliang@nudt.edu.cn

Feng Liu*
National University of Defense Technology
richardlf@nudt.edu.cn

Zhen Huang
National University of Defense Technology
huangzhen@nudt.edu.cn

Dongsheng Li
National University of Defense Technology
dsli@nudt.edu.cn

Abstract

In the realm of deep learning, many neural network-based models have emerged for recognizing argument relationships. However, these approaches often focus on specific argument-mining tasks aligned with their research goals and don't address all argument-mining subtasks comprehensively. Moreover, previous studies tend to assess models on a limited set of datasets from a single domain, sometimes relying on domain-specific external knowledge, which limits these methods' practical applicability. In this paper, we introduce LFAM, an end-to-end multi-task learning-based argument mining model. This model identifies argument component boundaries by using BIO labels at first. Then combines token-level and span-level features to capture robust contextual information. Finally, it employs the bi-affine structure for dependency parsing to represent argument relationships and uses the Chu-Liu/Edmonds Algorithm to predict argument relationship existence with maximal tree diagrams, while achieving argument relationship classification. We conducted experiments on six argument-mining datasets across various domains, demonstrating the effectiveness of LFAM.

1 Introduction

Argument Relation Recognition (ARR) is to identify semantic connections among key argumentative statements within argumentative texts, while simultaneously assessing their presence and categorizing them. ARR comprises two essential aspects in argument mining: Argument Relation Identification (ARI) and Argument Relation Classification (ARC). These tasks require a deep understanding of the complex semantic interactions among the textual components. As shown in Fig. 1, argumentation relationship recognition serves as a crucial step. For instance, when comparing Argument text 1 and Argument text 2, it's evident that they lack semantic correlation. On the other hand, Argument text 4 refines the ideas presented in Argument text 3, indicating a supportive relationship between them. Additionally, Argument text 6 directly counters the points made in Argument text 5, emphasizing the role of public transportation as a vital element in transportation choices.

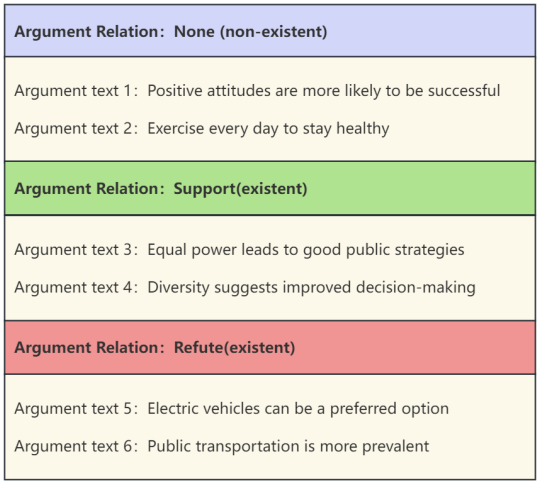

Figure 1. Argumentative Relationship Recognition Task.

With the rapid rise of deep learning, numerous neural network-based models have emerged in academic discussions, focusing on recognizing argument relationships. Scholars have extensively explored the connections between these tasks, proposing various multi-task learning approaches. For instance, Liao et al. [12] have introduced a multi-task iterative training approach that encompasses three subtasks

*Corresponding Authors

within the argument mining domain. In parallel, Dutta et al. [7] have advanced a pioneering prompt-based argument relation recognition framework, which seamlessly integrates pertinent contextual information, thereby engendering a model characterized by robust generalizability. These approaches often involve joint training of specific sub-tasks, each contributing distinct capabilities in the field of argument mining, ultimately leading to enhanced recognition performance.

Nonetheless, these methods exhibit inherent limitations. The pursuit of constructing a comprehensive end-to-end argument-mining model remains a challenging task because it requires solving all argument-mining subtasks at once in a unified framework while maintaining consistent performance. Researchers[15, 16] often choose to focus on specific argument-mining subtasks that align with their research goals for joint learning. This approach frequently overlooks a thorough examination of all argument-mining subtasks. Similarly, prevailing studies[24] typically assess model effectiveness using a limited set of datasets in a specific domain, and some researchers incorporate domain-specific external knowledge to improve the identification process. However, this practice may not be widely applicable in practical situations.

In light of these considerations, this paper introduces a novel argument-mining model grounded in an end-to-end multi-task learning model (LFAM). The model marks argument component boundaries through the application of BIO tags. Then it conducts distinct token-level and span-level feature interactions to capture richer contextual information. Finally, the architecture employs the Bi-affine structure for dependency parsing, thereby obtaining the representation of argument relations, and leveraging the Chu-Liu/Edmonds Algorithm to forecast the presence of argument relations through maximal tree diagrams, while achieving argument relation classification.

The main contributions of this paper are as follows: (1)We propose LFAM, a comprehensive end-to-end argument mining model that handles all argument mining tasks while ensuring stable and consistent performance. (2)We use the bi-affine structure and the Chu-Liu/Edmonds Algorithm for dependency parsing to model argument relationships. Additionally, we employ maximal tree diagrams to predict the presence and category of argument relationships. (3)We conducted experiments on six argument-mining datasets spanning various domains, illustrating the effectiveness of LFAM.

2 Related Work

The main goal of argument relation recognition is to find meaningful connections between important statements in argumentative texts while checking if these connections exist and sorting them into categories. Currently, the approaches for this task can be grouped into three categories:(1) textual feature-based methods, (2) shallow feature extraction methods, and (3) deep feature extraction methods.

2.1 Textual Feature-based Methods

Textual feature-based methods primarily rely on features crafted through human expertise to facilitate recognition, with these features including lexical, semantic, contextual, and core idea constructs. Mochales et al. [14] applied human-constructed, context-independent grammatical features to predict argument component relationships, aligning with grammatical rules mirroring the rhetorical and structural attributes commonly found in judicial texts. These textual features have been employed diversely in the literature. For instance, Persing et al. [22] extracted structural, grammatical, and lexical features for each argument, using positional information for structural attributes and character analysis for lexical features. Nguyen et al. [17] augmented their model's performance by combining topic-related information from the paper with features captured from a contextual window of sentences surrounding the argument. Paul et al. [20] introduced an unsupervised graph-based ranking approach, incorporating multi-hop knowledge from external sources into their model and utilizing a cross-attention mechanism [10] to enhance text and knowledge representations, leading to improved argumentation relation recognition across datasets. While these methods can improve performance in domain-specific contexts, they need manual feature extraction and setup, limiting their generalizability.

2.2 Shallow Feature Extraction Methods

Shallow feature extraction methods primarily employ classical early machine learning models. Researchers choose the different traditional machine-learning models based on the feature patterns found in various datasets. Subsequently, labeled data is constructed using these selected feature patterns to train the argument relation recognition model. Ghosh et al. [9] utilized supervised machine learning with ten-fold cross-validation for argument relation classification. They introduced a novel approach based on logistic regression and decision trees. Rosenthal et al. [25] conducted experiments with methods involving plain bayes, logistic regression, and decision trees, with their findings showing that the logistic regression-based classifier outperformed the others. Carstens et al. [5] proposed a relationship-based argument-mining method that initially classifies sentences based on their mutual relationships, offering a new perspective on argument-mining relationships. Stab et al. [26] proposed a solution using a traditional machine learning approach, employing a binary support vector machine for making predictions about relationships and classifying argument relationship types. Wang et al. [29] applied a Conditional Random Fields (CRF)-based sequential model for sentence-level or paragraph-level text prediction. However, methods relying on shallow feature

extraction are typically structurally simple, lack generalizability across diverse datasets, and exhibit suboptimal overall recognition performance, thereby limiting their widespread adoption.

2.3 Deep Feature Extraction Methods

Deep feature extraction methods employ deep modeling techniques based on pre-trained language models, often involving the utilization of pre-trained language models and frequently adopting multi-task learning approaches[6, 31]. This strategy is employed to significantly enhance the performance of recognizing argument relationships. Opitz et al. [18] utilized the BERT model as the foundational word vector and introduced relevant connectives based on various argument relationships to enrich the contextual information within the original argumentative text. They transform the task from predicting argument relations to assessing the logical coherence of sorted pairs of text. Experimental results underscored the method's effectiveness, especially in improving the accuracy of predicting objection-type argument relations. Liao et al. [12] proposed a multi-task iterative training method, addressing three sub-tasks related to the argument mining domain. Dutta et al. [7] introduced a novel prompt-based argumenta relation recognition method that integrates contextual information. This model possesses several attributes, including the ability to recognize argument relations in the domain of argument mining and high generalizability. Typically, these methods are jointly trained with specific subtasks within the argument mining domain to enhance recognition outcomes through complementary performance. However, it's important to note that these approaches may not include all four subtasks of argument mining, potentially limiting their broader applicability.

3 Problem Definition

LFAM is designed to tackle all fundamental subtasks within the realm of argument mining: (1) boundary segmentation, (2) component recognition, (3) relation identification, and (4) relation classification. Firstly, we use the token sequence $E = \{\omega_1, ..., \omega_n\}$ denote the thesis text, where ω_i signifies the i-th token in the thesis text. For the boundary segmentation task, we employ the notation $< s, e > \in \mathcal{S}$ to signify the range of selected argument components, where s and e represent the index of the starting and the ending token position of the argument component respectively, and $\mathcal{S}$ constitutes the set containing these argument components. For the component recognition task, we employ the triple notation $< s, e, c > \in C$, where c indicates the type of argument component associated with the $< s, e >$, and C represents a set of pre-defined argument component types. For the relation identification task, it can be expressed as a quintuple $< s_{src}, e_{src}, s_{tar}, e_{tar}, b > \in \mathcal{B}$, where s_{src} and e_{src} indicate the source argument component, s_{tar} and e_{tar} indicate the target argument component, the variable b denotes the presence or absence of a relationship, and $\mathcal{B}$ represents a bicategorical set that including the existence and non-existence of argumentative relations. Finally, the relation classification task is also denoted as a quintuple $< s_{src}, e_{src}, s_{tar}, e_{tar}, r > \in \mathcal{R}$, where r signifies the specific type of relationship between two argument components, and $\mathcal{R}$ denotes the set of predefined argument relation types.

4 Model

The LFAM framework comprises four essential components: (1) a text embedding layer, (2) a feature extraction layer, (3) a bi-affine layer, and (4) a maximal tree map layer. The text embedding layer leverages the semantic knowledge of a pre-trained model to encode each token within the original thesis text while employing BIO tags for boundary segmentation. The feature extraction layer employs Bidirectional Long Short-Term Memory (BiLSTM) to boost interactions at both the token level and span level, thus addressing the component identification subtask. The bi-affine layer employs Bi-affine operations to calculate the probability of a directed edge connecting any two spans. Subsequently, the maximum tree graph layer utilizes the maximum tree graph algorithm to construct an argument structure graph, for both relationship identification and relationship classification. Fig. 2 illustrates the framework of LFAM.

4.1 Text Embedding Layer

We employ the Longformer-base[3] pre-trained language model to encode the thesis texts, which is a suitable choice for processing token-level argumentative content. Given the initial input thesis text $E = \{\omega_1, \omega_2, ..., \omega_n\}$, we transform it into a textual representation $\mathbf{E} = \{token_1, token_2, ..., token_n\}$. To capture richer semantic knowledge, it is worth noting that the pre-trained language model predominantly stores semantic information in its higher-level network layers. Consequently, we concatenate the feature information from the last four layers of the model to create a representation for each token $token_i = [LF_i^{(9)}, LF_i^{(10)}, LF_i^{(11)}, LF_i^{(12)}] \in \mathcal{R}^{(4h)}$. Subsequently, we apply an MLP layer to reduce the dimensionality of each token to the standard dimension h. Additionally, an essential token "$[CLS]$" is the beginning of all tokens, serving as a global attention focal point for the Longformer model to capture comprehensive argumentation knowledge from the entire text.

Next, the boundary segmentation is addressed through the utilization of BIO labels, where "B" signifies the "Begin" of a span, "I" denotes an "Intermediate" position within the span, and "O" indicates other characters, typically with the first "O" adjacent to an "I" indicating the end of the span. To enhance the appropriateness of boundary segmentation, we refine BIO annotations by introducing the following rules: (1) if the initial label in the sequence is "I," it is modified to "B";

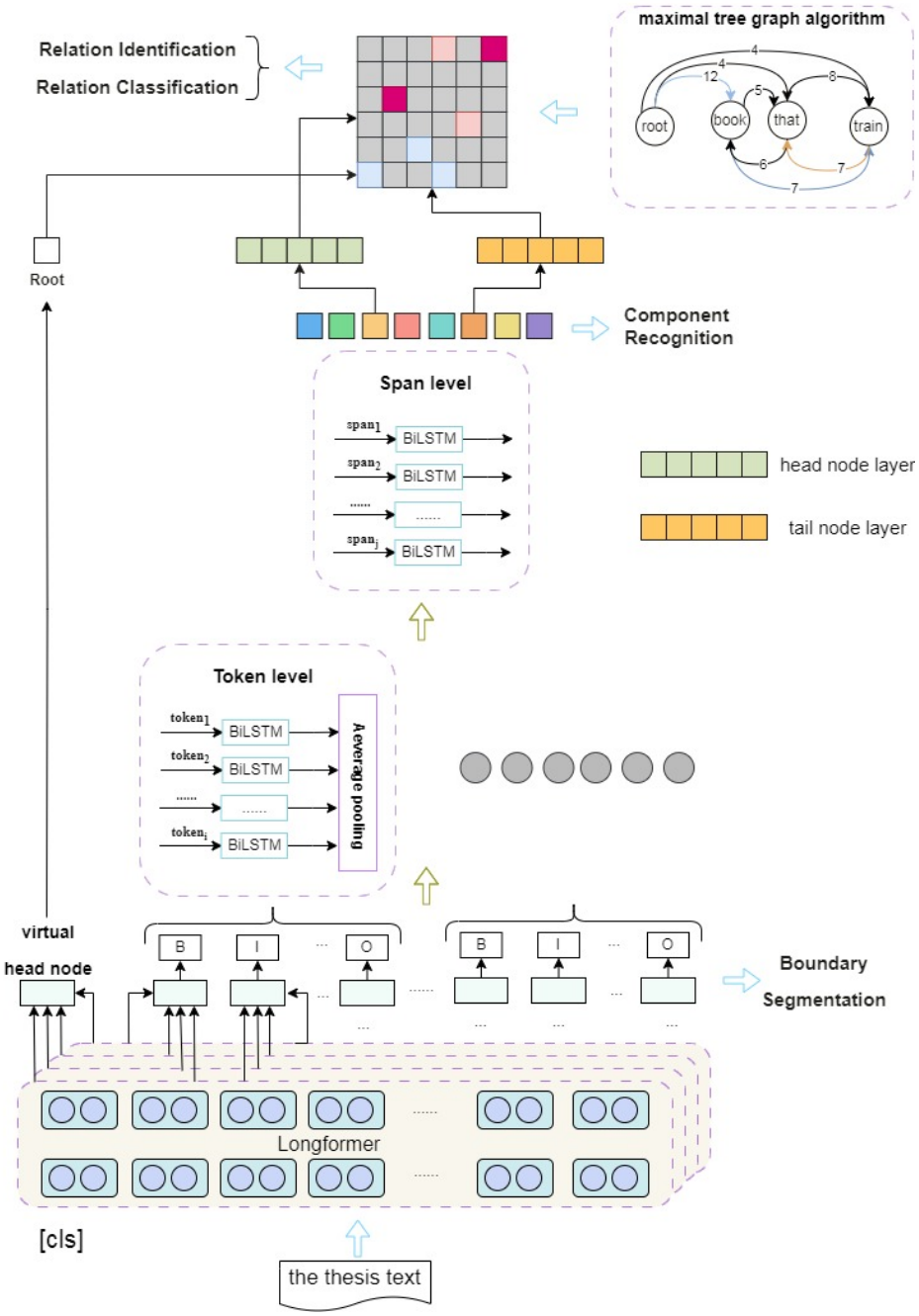

Figure 2. The overview of our LFAM framework.

(2) when the sequence begins with a non-labeled position and the current label is "I" while the previous one is "O," the current label is updated to "B." As illustrated in Fig. 3, in the case of the label "I" at ordinal number 4, it adheres to the optimization rule as the previous label is "O" and this "I" is the first one immediately following an "O", so it is modified to "B". Finally, three valid span divisions are obtained as a result of this process.

Consequently, from the vectorized textual representation $\mathbf{E}$, we obtain the subsequent representation for the argument span denoted as $Span_{ij} = (i, j)|i, j \in [0, n], i \leq j \in \mathcal{S}$, with $\mathcal{S}$ representing the set encompassing all possible spans.

4.2 Feature Extraction Layer

The feature extraction layer captures feature information at both token and span levels. Initially, token-level feature

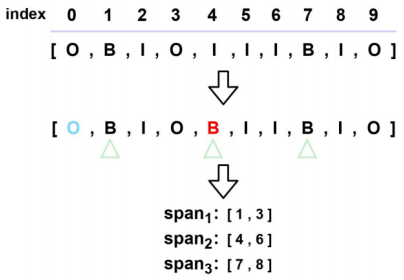

Figure 3. Optimized BIO annotation diagram.

interaction is accomplished using BiLSTM. The outputs from the bidirectional hidden layers are concatenated to form the feature representation for the input vectors at each time step. Subsequently, we compute the average of all token feature vectors within the span to obtain the feature representation for each span. The formula is as follows:

$$token_{h_t} = \overrightarrow{token}_{h_t} \oplus \overleftarrow{token}_{h_t} \tag{1}$$

$$\mathbf{span}_{ij} = \frac{token_i + ... + token_j}{j - i + 1}, \tag{2}$$

where $\overrightarrow{token}_{h_t}$ and $\overleftarrow{token}_{h_t}$ represent the feature output of the forward and backward hidden layers respectively, $\oplus$ denotes two square splice operations on two vector vectors, $token_{h_t}$ denotes the hidden layer feature output of the token at each time step t, and $span_{ij}$ denotes the span feature representation obtained after averaging each token feature vector within the span.

Subsequently, we conduct span-level feature interactions by employing BiLSTM as the foundational structure. We then reduce the interacted span features dimension to align with the number of argument component types present in the corresponding argument dataset and proceed to predict the output. The formula is as follows:

$$\mathbf{span}_{ij}^{BiLSTM} = BiLSTM(\mathbf{span}_{ij}) \tag{3}$$

$$component_{pred} = softmax(MLP(\mathbf{span}_{ij}^{BiLSTM})), \tag{4}$$

where $\mathbf{span}_{ij}^{BiLSTM}$ represents the span feature vector from the span-level feature interaction, and MLP denotes the function of dimensionality transformation. Finally, we obtain the probability of the corresponding argument components $component_{pred}$ by means of the softmax function.

4.3 Bi-affine Layer

The bi-affine layer employs a bi-affine structure primarily for parsing dependencies within argument information. Dependency syntactic parsing is centered on capturing the syntactic structure of text by examining the relationships among different components within a linguistic unit, providing a more precise understanding of text segment connections. Importantly, the bi-affine Layer offers faster training and inference speeds compared to conventional models relying on recurrent and convolutional neural networks. We apply the bi-affine structure to graph-based dependency syntactic analysis, which effectively addresses dependency issues between two nodes. It can predict the presence or absence of edges between two nodes and determine dependency edge labels, including the dependency relation and the dependency label. Typically, syntactic dependency parsing occurs within a sentence, but in argument text parsing, each argument span is treated as a node in the graph.

The initial step involves determining the presence of edges connecting nodes. If we consider an argument text with n argument component nodes, including a virtual head node, the total number of nodes is $d = n + 1$. For each node, we generate a score vector denoted as $s_i \in \mathbb{R}^{d \times 1}$. All argument component nodes can be organized into a scoring matrix of size $\mathbb{R}^{d \times d}$. Then we employ an MLP to reconfigure the argument component feature vector $\mathbf{span}_{ij}^{BiLSTM}$, which is obtained in the preceding section. The operation results in a vector $h_i^{(src-head)} \in \mathbb{R}^{k \times 1}$, where typically, k is of a small magnitude. Subsequently, this vector is inputted into the affine layer. In this way, we can effectively mitigate the risk of overfitting. The formula is as follows:

$$h_i^{(src-head)} = MLP^{(src-head)}(s_i) \tag{5}$$

$$h_j^{(tar-tail)} = MLP^{(tar-tail)}(s_j) \tag{6}$$

$$s_{ij}^{edge} = H_i^{(src-head)} W_s h_j^{(tar-tail)} + b_s \tag{7}$$

$$P_{ij} = f(s_{ij}), \tag{8}$$

where two MLPs are dedicated to the source node and the target node respectively, $H_i^{(src-head)} \in \mathbb{R}^{dxk}$ represents a matrix of feature vectors h, while these vectors are obtained by stacking the features from d argument components after applying quadratic encoding through an MLP, W_s is the weight matrix, b_s is the bias vector, $s_{ij}^{(edge)}$ is the score of the dependency edge between argument component i and argument component j. It is necessary to pass the dependent edge score s_{ij} into the nonlinear function f to obtain the probability distribution of dependent edges. It should be clarified that f is a $sigmoid$ function for determining the existence of dependent edges and f is a $softmax$ function for determining dependent labels.

This results in a probability distribution of dependency edges P_{exists}^d, where d represents the number of nodes. When assessing dependency labels, the quadratic encoding through MLP scales down to match the number of relationship types in the corresponding dataset and the subsequent steps remain consistent with the prior operations to obtain the distribution of dependency labels P_{label}^d.

4.4 Maximum Tree Layer

The maximal tree graph layer is designed to decode the graph representing the probability distribution of dependency edges obtained in the preceding subsection. This approach is inspired by the minimum tree graph method, which aims to identify a spanning tree within a directed weighted graph while designating a particular node as the root node to minimize the total sum of edge weights.

In the argument relations identification task, we represent the directionality of the argument relations through a directed graph. Therefore, each argument component serves as a node within the directed graph, and the directed edges between nodes represent argument relations. To keep semantics consistent, each argument node has a single directed edge pointing to other argument nodes. However, multiple argument nodes can point to the same argument node, meaning each node has only one outgoing edge but can receive multiple incoming edges.

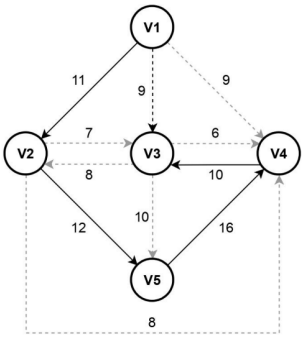

Figure 4. Maximum tree diagram.

Different from the conventional concept of minimal tree graphs based on original edge weights, where the smallest edge weights are selected to minimize usage costs. In our work, we extent incorporate the probability distribution of dependent edges. Consequently, we retain the directed edges with larger edge weights within the argument graph. As a result, the thesis text can be transformed into a tree graph. Fig. 4 shows an example of a directed graph producing a maximal tree graph. The five nodes in the graph and the dark edges between them form a basic maximal tree graph, of which node V_1 is the root node. There is only one path to the other nodes from V_1 and the sum of the numbers on all the dark-colored paths is also the maximum that can be generated for the entire graph.

We apply the Chu–Liu/Edmonds algorithm[28], known for its time complexity of $O(VE)$ and its status as the fastest minimal tree graph algorithm available. Utilizing the maximum tree diagram algorithm, we analyze the dependency edge probability distribution P^{d}_{exist} to derive the corresponding G^{tree}_{exist}. Subsequently, we discard connecting edges that start or end at the virtual root node to obtain the predicted argument relations. These predictions are based on the distribution of dependent labels P^{d}_{labels}, which select the argument relationship with the highest probability.

4.5 Loss Function

For the four subtasks of argument mining, we use a combination of multiple task loss functions for model parameter optimization, the loss function is defined as follows:

$$\mathcal{L} = \lambda_s \mathcal{L}_s + \lambda_c \mathcal{L}_c + \lambda_b \mathcal{L}_b + \lambda_r \mathcal{L}_r \tag{9}$$

where $\mathcal{L}_i (i \in s, c, b, r)$ are the loss functions corresponding to the boundary segmentation, component recognition, relation identification, and relation classification subtasks, respectively. $\lambda_i (i \in s, c, b, r)$ are the hyperparameters corresponding to the loss function for each subtask.

5 Experiment

5.1 Datesets

In order to validate the adequacy of LFAM, we conduct experiments in five benchmark datasets from multi-domain: AAEC-Essay[27], AAEC-Para[8], AASD[1], MTC[21], CDCP[19] and AbstRCT[13]. The detail of the datasets in App. A and the detail about the experiment setup in App. B.

5.2 Baselines

To evaluate the performance of LFAM, we compare LFAM with the following methods:

- **ILP-Joint**[27] is a joint learning model based on integer linear programming.
- **BLCC(BiLSTM-CRF-CNN)**[8] is an analytic model of argument structure based on sequence labeling.
- **LSTM-ER**[8] is an end-to-end model based on tree LSTM.
- **BiPAM-syn**[30] is an argument mining model based on pre-trained language models using syntactic information augmentation.
- **Cross-ST**[16] is an incremental learning approach based on argumentation datasets, and for the sake of fairness of comparison nature, only the experimental results of a single dataset are selected.
- **Pointer-Net**[11] utilizes a pointer network for the prediction of argument components and relationships.
- **Span-LSTM**[23] is a model that combines textual input sequences and argumentative markers for encoding.
- **Bert-Trans**[2] is an argument mining model based on neural transitions of parser states.
- **TSP**[15] is a graph-based non-tree-structured argument parsing and bisimulation of attention struct
- **Rel.RoBERTa**[13] is an argumentation model based on the RoBERTa parser.
- **Rel.SciBERT**[13] is an argumentation model based on the SciBERT parser.

Dataset	Method	Boundary	Component		Link	Relation	
			Micro	Macro		Micro	Macro
AAEC-Essay	BLCC	-	63.23	-	-	34.82	-
	LSTM-ER	-	66.21	-	-	29.56	-
	Cross-ST	85.21	**75.54**	**66.59**	55.66	55.17	42.30
	LFAM	**85.23**	75.21	66.07	**55.84**	**55.36**	**44.16**
AAEC-Para	BLCC	-	66.69	-	-	39.83	-
	LSTM-ER	-	70.83	-	-	45.52	-
	BiPAM-syn	-	73.5	-	-	46.4	-
	Cross-ST	-	**76.48**	-	-	**59.55**	-
	LFAM	85.55	75.51	66.13	56.38	55.52	44.57
AASD	Cross-ST	87.10	69.06	58.06	54.82	49.83	42.10
	LFAM	**87.23**	**71.51**	**59.44**	**59.80**	**53.99**	**46.45**
MTC	Cross-ST	**87.68**	**78.83**	73.77	53.43	45.92	33.07
	LFAM	86.70	78.80	**74.70**	**55.47**	**48.66**	**38.73**
CDCP	Cross-ST	82.88	**68.90**	**65.78**	31.94	31.94	16.26
	LFAM	**83.13**	68.13	64.63	**33.16**	**33.16**	**16.87**
AbstRCT	Cross-ST	70.29	64.16	**45.04**	**39.35**	**38.38**	31.91
	LFAM	**70.33**	**64.40**	44.90	39.02	38.04	**33.67**

Table 1. Experiment result of subtasks containing boundary delineation on Datasets.

5.2.1 Experimental Results of End-to-end Modeling.

Our experiments are based on the results of ten random number seeds taking the average, with the best experimental results bolded. Tab.1 shows a comparison of the performance of the baseline model and LFAM on the six datasets when the boundary segmentation subtask is included, as can be found from the experimental results:

(a) The LFAM model performs better than the best existing model in most of the evaluation measures across five datasets: AAEC-Essay, AASD, MTC, CDCP, and AbstRCT. Specifically, we see the biggest improvements in the three evaluation measures for relationship identification and classification, which are 4.98%, 4.16%, and 5.66%, respectively. On the AASD dataset, our LFAM model even outperforms the baseline model in all six evaluation measures, demonstrating its wide-ranging effectiveness.

(b) Despite the challenge of errors transferring between the four argument mining tasks, where how well you do in one task affects the others, and performance tends to decrease as you go along, we found something interesting. When the performance of the boundary task and the constituent task are similar, the performance of the LFAM model does not decrease obviously from recognizing constituents to recognizing relations compared to other models. This advantage helps it to achieve better performance in predicting argument relations.

(c) The performance on the AAEC-Para dataset is not ideal, doing better than models like BiPAM-syn but falling short of the Cross-ST model. This situation can be attributed to the fact that the AAEC-Para dataset is derived from a segmented version of the AAEC-Essay dataset, which is typically shorter. Consequently, the LFAM model's robust feature retrieval and extraction capabilities, may not be as effective in this condition

Dataset	Method	Component		Link	Relation	
		Micro	Macro		Micro	Macro
AAEC-Para	ILP-Joint	-	82.6	58.5	-	-
	Pointer-Net	-	84.9	60.8	-	-
	Span-LSTM	-	85.7	67.8	-	-
	Bert-Trans	-	**88.4**	**70.6**	-	-
	Cross-ST w.map	88.40	86.82	69.33	68.14	57.11
	Cross-ST	**87.44**	**79.55**	67.16	66.29	**55.84**
	LFAM	87.08	79.47	**68.65**	**67.49**	55.63
AASD	Cross-ST	77.78	65.90	63.96	58.30	49.22
	LFAM	**80.71**	**69.04**	**69.47**	**64.12**	**53.93**
MTC	ILP-Joint	-	85.7	48.6	-	-
	Pointer-Net	-	81.3	57.7	-	-
	Span-LSTM	-	83.5	57.5	-	-
	Cross-ST w.map	95.65	**93.08**	**65.06**	57.89	57.34
	Cross-ST	89.85	83.70	65.06	55.23	39.27
	LFAM	**91.22**	**86.07**	**69.45**	**60.69**	**46.19**
CDCP	TSP	-	78.91	34.04	-	-
	Bert-Trans	-	**82.5**	37.3	-	-
	Cross-ST	**81.03**	82.34	40.15	40.11	20.39
	LFAM	80.04	78.54	**41.81**	**41.74**	**21.20**
AbstRCT	Rel.RoBERTa	-	-	-	48.72	17.53
	Rel.SciBERT	-	-	-	**58.21**	36.76
	Cross-ST	89.37	67.57	59.65	57.10	47.12
	LFAM	**89.90**	**68.32**	**60.36**	57.73	**49.00**

Table 2. Performance comparison of the boundary delineation subtask without boundary delineation on Datasets.

5.2.2 Experimental Results of Non-End-to-end Modeling.

In addition, to compare the task performance with other non-end-to-end models, this section uses the golden result on the first subtask task boundary segmentation by directly using the golden result as the boundary segmentation and conducting experiments on the next three subtasks. Tab. 2 demonstrates the performance comparison between the baseline model and LFAM on the five datasets without boundary segmentation subtasks, as can be found from the experimental results:

(a) The LFAM model performs better than other models in most metrics on the AASD, MTC, CDCP, and AbstRCT

Dataset	Method	Boundary	Component		Link	Relation	
			Micro	Macro		Micro	Macro
AAEC-Essay	LFAM	**85.23**	**75.21**	**66.07**	**55.84**	**55.36**	**44.16**
	LFAM w/o. tree	84.28	74.93	65.09	54.54	54.07	42.30
AAEC-Para	LFAM	**85.55**	**75.51**	**66.13**	**56.38**	**55.52**	**44.57**
	LFAM w/o. tree	84.76	74.78	65.01	55.49	54.80	42.15
AASD	LFAM	**87.23**	71.51	59.44	**59.80**	**53.99**	**46.45**
	LFAM w/o. tree	87.21	**72.63**	**59.93**	54.87	50.61	44.21
MTC	LFAM	**86.70**	**78.80**	**74.70**	**55.47**	**48.66**	**38.73**
	LFAM w/o. tree	86.15	77.97	73.46	54.39	47.46	37.06
CDCP	LFAM	**83.13**	68.13	64.63	**33.16**	**33.16**	**16.87**
	LFAM w/o. tree	82.71	**68.31**	**64.90**	32.18	32.04	16.33
AbstRCT	LFAM	**70.33**	**64.40**	**44.90**	**39.02**	**38.04**	**33.67**
	LFAM w/o. tree	68.76	62.82	43.89	35.01	33.99	30.03

Table 3. Impact of maximum tree layers on LFAM Models.

datasets. Although it falls short of the baseline model in two metrics within the CDCP dataset's component recognition, we see improvements in all six metrics for the AASD, MTC, and AbstRCT datasets. This shows that the LFAM model is quite effective across a range of situations.

(b)In the AAEC-Para dataset, many researchers simplified the process by combining "Claim: Against" and "Claim: For" into a single "Claim" label during training and prediction, which we call "w. maps" in this section. Specifically, the Cross-ST w. map model performs worse than the Bert-Trans model in both component macro and link evaluation metrics. However, when we analyze a more detailed analysis of argument components, the LFAM model performs better than the Cross-ST model in the macro metrics for relationship identification and relationship recognition but is slightly behind in the other three metrics. Since the LFAM and Bert-Trans models use different architectures, the former is based on graph parsing and the latter on parser transformations. It suggests that combining these approaches in the future for argument mining tasks might lead to improved results in parsing argumentative texts.

(c)In the MTC dataset, we first changed the labels of argument parts to "statement" and "premise" categories, and we mapped argument connections to "support" and "attack" categories. In this setup, the Cross-ST w. map model performed the best for these two contrasting categories. But when we looked at the original labels, the LFAM model performs much better than the Cross-ST model in all aspects, showing that the LFAM model is effective on the MTC dataset.

To sum it up, the LFAM model performs really well compared to both the end-to-end and non-end-to-end models, especially in tasks like relationship identification and relationship classification in argument recognition. We further demonstrate the superiority of the LFAM model in the task of argument relationship classification in the App. **??**.

5.3 Ablation Study

Multi-task learning can be really helpful in handling different argument-related tasks together as seen in previous studies. In this part, we're looking at how having maximal tree layers affects the argument mining task. Tab. 3 shows the impact of removing these layers on the argument mining task across five datasets. The results are quite clear: when it comes to identifying and classifying argument relationships, all three performance metrics drop noticeably, with the largest decreases being 4.93%, 4.05%, and 3.63%, respectively. This confirms that having maximal tree layers is crucial for recognizing argument relationships. For the tasks of boundary segmentation and component recognition, things are a bit different. Only some metrics in the CDCP dataset show small improvements, while the other four datasets see a decline in performance. This can to some extent support the performance assistance of the maximum tree layer for the first two subtasks, and also demonstrate the advantages of multi-task learning.

5.4 Case Study

To further validate the effectiveness of the LFAM model, a typical case from the MTC dataset is selected for analysis in this section, the detail is in App. C.

6 Conclusion

In this paper, we focus on the argument mining task, especially on argument relation recognition. Aiming at the lack of a complete end-to-end model and the lack of validation of model generalization in multiple domain datasets in existing methods, we propose an end-to-end multi-task learning-based argument relation recognition model LFAM. LFAM incorporates the semantic information of the pre-trained model well, provides a suitable deconstruction of the four subtasks of argument mining, and the maximal tree graph provides good performance of the argument relation identification and argument relation identification classification. The experimental results show that the LFAM model exhibits excellent performance and wide application potential on six argument mining datasets in multiple domains.

References

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

[2] Jianzhu Bao, Chuang Fan, Jipeng Wu, Yixue Dang, Jiachen Du, and Ruifeng Xu. 2021. 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)*.

[3] Iz Beltagy, MatthewE. Peters, and Arman Cohan. 2020. Longformer: The Long-Document Transformer. *arXiv: Computation and Language,arXiv: Computation and Language* (Apr 2020).

[4] Michael W Browne. 2000. Cross-validation methods. *Journal of Mathematical Psychology* 44, 1 (Mar 2000), 108–132.

[5] Lucas Carstens and Francesca Toni. 2015. Towards relation based Argumentation Mining. In *Proceedings of the 2nd Workshop on Argumentation Mining*.

[6] Michael Crawshaw. 2020. Multi-Task Learning with Deep Neural Networks: A Survey. (Sep 2020).

[7] Subhabrata Dutta, Jeevesh Juneja, Dipankar Das, and Tanmoy Chakraborty. [n. d.]. Can Unsupervised Knowledge Transfer from Social Discussions Help Argument Mining? ([n. d.]).

[8] Steffen Eger, Johannes Daxenberger, and Iryna Gurevych. 2017. Neural End-to-End Learning for Computational Argumentation Mining. In *Annual Meeting of the Association for Computational Linguistics*.

[9] Debanjan Ghosh, Smaranda Muresan, Nina Wacholder, Mark Aakhus, and Matthew Mitsui. 2014. Analyzing Argumentative Discourse Units in Online Interactions. In *Proceedings of the First Workshop on Argumentation Mining*.

[10] Ruibing Hou, Hong Chang, Bingpeng Ma, Shiguang Shan, and Xilin Chen. 2019. Cross Attention Network for Few-shot Classification. *Neural Information Processing Systems,Neural Information Processing Systems* (Sep 2019).

[11] 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*.

[12] XW Liao, ZZ Chen, L Gui, XQ Cheng, and GL Chen. 2019. An argumentation mining method based on multi-task iterative learning. *Chinese Journal of Computers* 42, 7 (2019), 1524–1538.

[13] Tobias Mayer, Elena Cabrio, and Serena Villata. 2020. Transformer-based Argument Mining for Healthcare Applications. (Aug 2020).

[14] Raquel Mochales and Marie-Francine Moens. 2011. Argumentation mining. *Artificial Intelligence and Law* (Mar 2011), 1–22.

[15] 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*.

[16] 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* (May 2022), 639–658.

[17] Huy Nguyen and Diane Litman. 2016. Context-aware Argumentative Relation Mining. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

[18] Juri Opitz. 2019. Argumentative Relation Classification as Plausibility Ranking. *KONVENS,KONVENS* (Sep 2019).

[19] Joonsuk Park and Claire Cardie. 2018. A Corpus of eRulemaking User Comments for Measuring Evaluability of Arguments. *Language Resources and Evaluation,Language Resources and Evaluation* (May 2018).

[20] Debjit Paul, Juri Opitz, Maria Becker, Jonathan Kobbe, Graeme Hirst, and Anette Frank. 2020. Argumentative Relation Classification with Background Knowledge. (Jan 2020).

[21] Andreas Peldszus and Manfred Stede. 2015. An annotated corpus of argumentative microtexts. In *Argumentation and Reasoned Action: Proceedings of the 1st European Conference on Argumentation, Lisbon*, Vol. 2. 801–815.

[22] 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*.

[23] Peter Potash, Alexey Romanov, and Anna Rumshisky. 2016. Here's My Point: Joint Pointer Architecture for Argument Mining. *arXiv: Computation and Language,arXiv: Computation and Language* (Dec 2016).

[24] Ellen Riloff, David Chiang, Hockenmaier Julia, and Tsujii Jun'ichi. 2018. Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing. (Jan 2018).

[25] Sara Rosenthal and Kathy McKeown. 2015. I Couldnâ€™t Agree More: The Role of Conversational Structure in Agreement and Disagreement Detection in Online Discussions. In *Proceedings of the 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue*.

[26] Christian Stab and Iryna Gurevych. 2014. Identifying Argumentative Discourse Structures in Persuasive Essays. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

[27] Christian Stab and Iryna Gurevych. 2016. Parsing Argumentation Structures in Persuasive Essays. *Computational Linguistics,Computational Linguistics* (Apr 2016).

[28] R. E. Tarjan. 1977. Finding optimum branchings. *Networks* 7, 1 (Jan 1977), 25–35.

[29] Lu Wang and Claire Cardie. 2014. Improving Agreement and Disagreement Identification in Online Discussions with A Socially-Tuned Sentiment Lexicon. In *Proceedings of the 5th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*.

[30] 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*.

[31] Yingxiu Zhao, Zhiliang Tian, Huaxiu Yao, Yinhe Zheng, Dongkyu Lee, Yiping Song, Jian Sun, and NevinL. Zhang. [n. d.]. Improving Meta-learning for Low-resource Text Classification and Generation via Memory Imitation. ([n. d.]).

A Datasets

The statistics of the three datasets are shown in Tab. 4, where the AAEC-Essay dataset and the AAEC-Para dataset are denoted together as the AAEC dataset.

Considering the small amount of data in the AASD and MTC datasets, which are not divided into fixed training and testing sets, five-fold cross-validation[4] is applied to the training data to iterate the model parameters to ensure the reliability of the experiments, and the AAEC, CDCP, and AbsRCT datasets are processed in the normal training way.

B Experimental Setup

In the experiments, the model is optimized using the Adam optimizer, and a linear hot-start mechanism is adopted. The batch size is set to 4, the dropout is set to 0.1, the hidden layer dimension is set to 768, the number of training epochs

Dataset	Argumentative Component/Quantity						Argumentative Relationships / Quantity				
AAEC	Major claim 751		Claim: For 1228		Claim: Against 278	Pre. 3932	Support 3613			Attack 219	
AASD	Pro. 110	Ass. 88	Res. 73	Obs. 11	Mea. 63	Des. 7	Sup. 124	Att. 0	Det. 130	Add. 27	Seq. 11
MTC	Proponent 451			Opponent 125			Sup. 263	Exa. 9	Reb. 108	Undercut 63	
CDCP	Policy 815	Value 2160	Fact 746	Testimony 1026		Ref. 32	Reason 1307			Evidence 46	
AbstRCT	Major Claim 93		Claim 993		Evidence 2193		Sup. 1762	Att. 60	Partial-Attack 238		

Table 4. The label distribution statistics for the datasets.

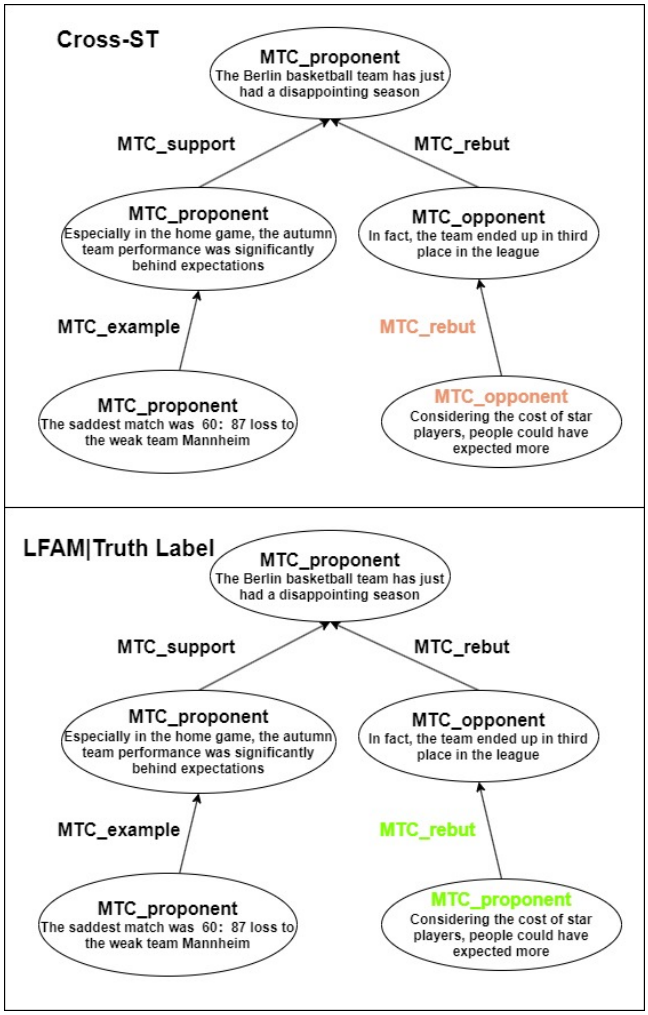

Figure 5. Case study.

is set to 30, the validation is performed every 2 epochs, and the learning rate is set to $1e-5$. The hyperparameters $\lambda_i (i \in s, c, b, r)$ of the four sub-task loss functions of argument mining were selected, and the values of the hyperparameters were varied for each dataset.

we use two evaluation metrics, Macro-F1 and Micro-F1. Macro-F1 calculates the F1 scores under each category of labels separately and takes the average value, while Micro-F1 considers all labels uniformly and counts the F1 scores of all labels directly. The Macro-F1 score is used for all four subtasks, and the Micro-F1 score is additionally used for the two key subtasks of component identification and relationship classification.

C Case Study

Fig. 5 shows the visualization of the argument structure of the data under the two models and the real labels. Each ellipse represents a component of the argument, and the connecting arrows indicate the relationship between the two components of the argument. It is clear that the Cross-ST model incorrectly predicts the type of argument component "but given the cost of the star player, one would have expected more" and the type of argument relationship with "the team ended up in third place" as shown by the orange color in the figure above. The LFAM model corrects for these types of arguments, as shown in orange in the figure above. The LFAM model corrects these confusing argument structures by determining that there should be an undercut relationship between the two argument components that refutes but does not completely negate the argument and that the argument component in the lower right corner should belong to the supporter proponent, as shown in green in the figure below.