Improving Synthetic Data Generation for Argument Mining via Adversarial Reinforcement Learning

Zhijun Zhang[1], Qianlong Wang[2,*], Keyang Ding[3], Genan Dai[2], Bowen Zhang[2], Bin Liang[4], Ruifeng Xu[3,5], Yongsheng Liang[1,2,3]
[1]College of Applied Science, Shenzhen University, Shenzhen, China
[2]School of Artificial Intelligence, Shenzhen Technology University, Shenzhen, China
[3]Harbin Institute of Technology (Shenzhen), Shenzhen, China
[4]The Chinese University of Hong Kong, Hong Kong, China
[5]Pengcheng Laboratory, Shenzhen, China
2510263003@mails.szu.edu.cn, wangqianlong@sztu.edu.cn

Abstract

Argument Mining (AM) is fundamentally constrained by the scarcity of high-quality structure-annotated datasets. While LLMs have shown promise in synthetic data generation, producing synthetic AM data that is both structurally accurate and sufficiently diverse remains a challenging problem. To address this problem, we revisit synthetic data generation for AM from a new perspective and propose a novel *adversarial reinforcement learning framework for data synthesis*. The proposed framework jointly optimizes the generator and the discriminator in an adversarial loop, in which the generator produces structured AM instances, and the discriminator provides learning signals by distinguishing real data from synthetic candidates. This enables the generator to progressively improve both the structural accuracy of generated argument data while maintaining diversity through adversarial feedback. Extensive experiments demonstrate that the proposed framework consistently improves AM performance on three benchmark datasets in both full-data and low-resource settings, validating its effectiveness and scalability.

1 Introduction

Argument Mining (AM) aims to identify argument structures in natural language text automatically[1]. Specifically, it involves detecting argumentative components, such as claims and premises, and identifying the relations among them, including support and attack relations (Lippi and Torroni, 2015). As one of the core tasks in computational argumentation, AM is highly significant for downstream applications such as argument retrieval, opinion analysis, and argument generation (Wachsmuth et al., 2017; Lawrence and Reed, 2019; Chen et al., 2024).

However, the development of end-to-end AM models has long been constrained by data scarcity (Dutta et al., 2022; Morio et al., 2022). This is because AM requires not only accurately determining whether a text span is argumentative, but also performing fine-grained type classification of components and relation mapping at the structural level. Such high complexity makes the manual construction of high-quality AM corpora usually costly (Das and Stede, 2018). As a result, existing benchmark datasets for AM are typically small in scale despite their research value, which limits the training of robust and high-performing AM models. For instance, the AAEC dataset (Stab and Gurevych, 2017) contains only 289 training essays.

In recent years, the development of large language models (LLMs) has provided new possibilities for alleviating data scarcity in AM. Although LLM-based synthetic data generation has shown remarkable effectiveness in improving the performance of various NLP tasks (Tan et al., 2024; Dai et al., 2025), generating synthetic data for AM remains highly challenging. This is because high-quality AM samples should not only be natural and fluent at the linguistic level, but also preserve structurally valid argumentative relations and component organization (Persing and Ng, 2016). Moreover, the AM samples should maintain sufficient diversity in both topics and argument structures. Recent studies have begun exploring this direction. In particular, Bao et al. leveraged prompt engineering with GPT-4o to enhance both the quality and diversity of synthetic AM data. Despite promising gains, this approach still relies heavily on costly closed-source LLMs and treats quality and diversity in separate generation pipelines, which makes it difficult to produce synthetic samples that are both structurally reliable and sufficiently diverse.

Building upon these limitations, we argue that synthetic data generation for AM should jointly optimize data quality and diversity within a unified framework. A straightforward solution is to employ open-source LLMs with stochastic sam-

* Corresponding author.
[1]The task definition of AM can be found in Appendix A.

Findings of the Association for Computational Linguistics: EMNLP 2026, pages 20017–20036
October 24-29, 2026 ©2026 Association for Computational Linguistics

pling strategies to increase output diversity while generating high-quality synthetic data. However, sampling and generating without explicit supervision signals often lead to noisy instances. In this context, reinforcement learning may be considered as a natural mechanism to incorporate some signals and optimize the generation process toward both quality and diversity, thereby encouraging outputs that more closely resemble real data.

Motivated by this, we approach synthetic data generation for AM from a new perspective and propose a novel *Adversarial Reinforcement Learning framework for Data Synthesis*. Our key idea is to jointly train two LLMs in an adversarial manner: the generator and the discriminator, where reinforcement learning is used to propagate the discriminator's feedback as learning signals for the generator. Specifically, the generator first samples some synthetic AM data conditioned on gold-annotated data. The discriminator is then presented with a candidate set containing one annotated sample and multiple synthetic samples, and is required to identify the real instance. The discriminator's decisions are converted into reward signals and further combined with rule-based constraints to form the final training signal. Here, we optimize the generator using Group Relative Policy Optimization (GRPO) (Shao et al., 2024), enabling it to produce increasingly realistic and diverse synthetic data. Meanwhile, the generator is encouraged to produce challenging samples that the discriminator cannot reliably distinguish from real annotated data. Subsequently, the improved generator produces more diverse and harder-to-detect synthetic samples, which are then used to further refine the discriminator's ability. Here, the discriminator is trained to maximize its identification accuracy. Through this alternating adversarial optimization process, both LLMs improve iteratively. As a result, the generator progressively learns to produce synthetic AM data with higher linguistic naturalness, more accurate argument structures, and greater diversity.

We conduct comprehensive experiments on three widely used AM datasets. Empirical results show that augmenting the original training data with synthetic samples generated by our framework (implemented with Qwen3-8B) consistently yields substantial improvements over strong AM baselines across both full-data and low-resource settings. Notably, our framework even achieves performance surpassing data generated by stronger proprietary systems such as GPT-4o. Further analysis confirms that the gains stem from both improved linguistic naturalness and greater structural diversity of the generated data.

2 Related Work

2.1 Argument Mining

AM aims to identify argumentative content and its internal structure in unstructured text (Lawrence and Reed, 2019). Argument structures involve argument components, component types, and logical relations between components. Thus, existing studies decompose the task into several more tractable subtasks, including argument component segmentation and classification, which identify component spans in text (Moens et al., 2007; Stab and Gurevych, 2017; Lippi and Torroni, 2015), and argumentative relation identification, which determines whether semantic relations exist between components (Cocarascu and Toni, 2017; Jo et al., 2021).

Such staged modeling approaches are susceptible to error propagation. Consequently, growing attention has recently been devoted to joint modeling and end-to-end AM methods (Persing and Ng, 2016; Eger et al., 2017). Early methods mainly drew on natural language parsing techniques, such as sequence labeling (Wang et al., 2021; Ye and Teufel, 2021) or logical inference problems (Bao et al., 2021; Morio et al., 2022). With the development of LLMs, AM has gradually been reformulated as a seq2seq generation task (Kawarada et al., 2024; Sun et al., 2024; Bao et al., 2025b).

However, AM models are still constrained by the scarcity of annotated corpora that limits further performance improvements. This is mainly due to the inherent complexity of argument structures, making data annotation highly costly.

2.2 Synthetic Data Generation

Synthetic data generation has become an important approach for data augmentation in NLP, motivating researchers to leverage LLMs to construct task-specific training samples (Bao et al., 2023; Wang et al., 2023; Guo and Chen, 2024; Havrilla et al., 2024). Compared with general generations, synthetic data generation for structured-output tasks needs to ensure both text quality and the correctness of label structures. Thus, task-specific constraints and structure-preserving mechanisms are needed to control the quality of synthetic samples (Josifoski et al., 2023; Zhang et al., 2024).

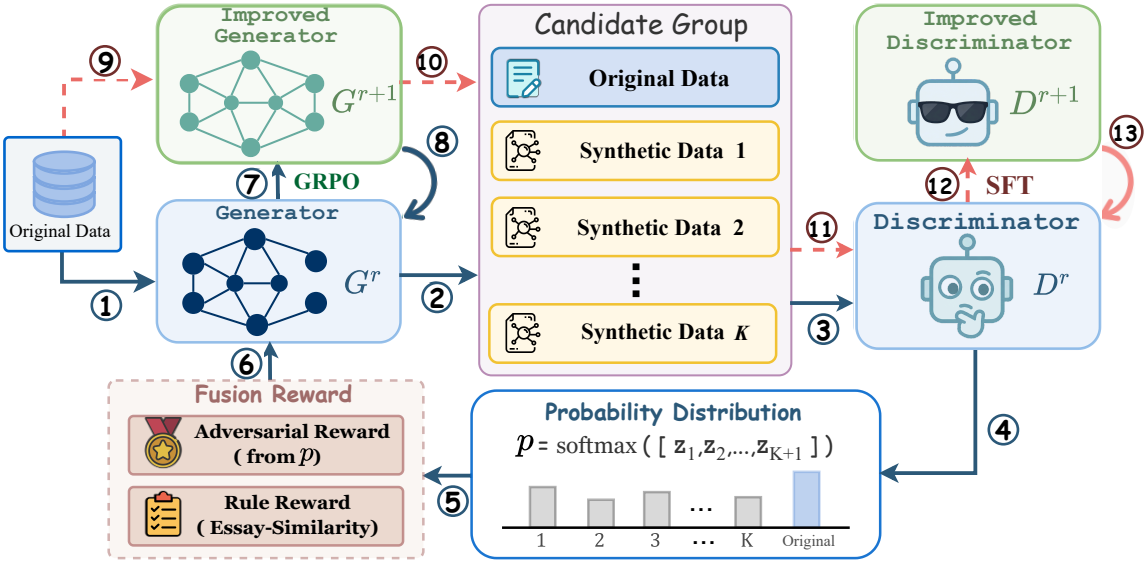


Figure 1: Overview of the proposed framework. The framework consists of two iterative stages: generator optimization (①–⑧) and discriminator refinement (⑨–⑬). Through alternating adversarial training, the generator and discriminator progressively improve together.

Recent studies have explored generating AM data with GPT-4o from both quality and diversity perspectives, showing that synthetic data can improve AM model performance (Bao et al., 2025a). However, existing GPT-4o-based approaches are costly and rely heavily on prompt engineering, making data quality and diversity difficult to control without model adaptation.

In recent years, reinforcement learning and preference optimization have also been increasingly applied to synthetic data generation, where prior studies optimize the data generation process using preference signals or downstream feedback (Wang et al., 2024; Hou et al., 2025; Zweiger et al., 2026; Wang et al., 2026a,b). However, these methods mainly target general data generation scenarios and are not specifically designed to maintain consistency between argumentative text and structured AM annotations. This work addresses this research gap by exploring the application of reinforcement learning to AM data synthesis.

3 Framework

3.1 Overview

As illustrated in Figure 1, our framework consists of two iterative stages: generator optimization and discriminator refinement. The generator G_θ first produces multiple synthetic AM samples based on an annotated instance, and is optimized using reward signals derived from the discriminator's feedback. The discriminator D_ϕ is then further trained to distinguish annotated samples from newly generated synthetic samples. These two stages are alternately repeated for several rounds, enabling both models to progressively improve through adversarial interaction.

3.2 Generator Optimization Stage

Before adversarial training, we initialize the generator to provide a basic ability to produce valid AM synthetic samples. Let the annotated AM dataset be $\mathcal{D} = \{z_i\}_{i=1}^{N}$, where each sample $z_i = (x_i, y_i)$ consists of an input text x_i and its corresponding structured argument annotation y_i.

Generator initialization. Given an annotated sample z, we serialize it into an instruction-style prompt u. The target output v is a synthetic AM sample that follows the same annotation schema while differing from the annotated sample in both textual content and argument structure. The generator is initialized through supervised fine-tuning (SFT) with the following objective:

$$\mathcal{L}_{\text{SFT}}(\theta) = -\mathbb{E}_{(u,v)\sim z, z\sim\mathcal{D}} \log p_\theta(v \mid u) \quad (1)$$

Here, u denotes the prompt constructed from the real AM sample x, and v denotes the corresponding target synthetic sample[2]. During this stage, the generator learns to produce complete text-annotation pairs that follow the target AM schema.

Reinforcement learning. After initialization, the generator is further optimized through adversarial

[2]Initially generated using Qwen3-8B with a detailed prompt, followed by basic error filtering.

reinforcement learning using feedback from the discriminator. Specifically, for each annotated sample z, the generator produces K synthetic samples. We then construct a shuffled candidate set C consisting of one annotated sample and K synthetic samples. Let y_{position} represent the position of the annotated sample. To encourage the generator to produce high-quality samples that can confuse the discriminator, the discriminator reward is defined:

$$R_{\text{dis}} = \min\left(-\log\left(p_\phi(y_{\text{position}} \mid C) + \epsilon\right), c\right) \tag{2}$$

where ϵ is a small constant for numerical stability, and c is the reward clipping threshold. Here, a larger reward indicates that the discriminator finds it more difficult to identify the annotated sample, suggesting that the generated samples are closer to high-quality real AM data. The clipping operation prevents excessively large rewards that may destabilize training.

In preliminary experiments, we find that the generator sometimes makes only shallow modifications to the source sample, or even directly reuses expressions from the original data. To alleviate this issue, we introduce a source similarity penalty that discourages generated samples from being overly similar to the source sample. Specifically, we compute the ROUGE-L F1 similarity between the synthetic text and the source text. Here, if the similarity is below a threshold τ, no penalty is applied. Otherwise, the penalty increases with similarity and is bounded within $[-1, 0]$. Let R^k_{sim} denote the similarity penalty for the k-th synthetic sample. The final reward is defined as:

$$R_k = R_{\text{dis}} + \lambda_1 \times R^k_{\text{sim}} \tag{3}$$

Here, λ_1 is the weight of the similarity penalty. This reward encourages the generator to produce synthetic samples that are more realistic and difficult to distinguish, while the similarity penalty explicitly promotes sample novelty and diversity.

After obtaining the final rewards, we optimize the generator using GRPO, which estimates the relative advantage of each synthetic sample based on the rewards of all outputs within the same group. Let A_k denote the advantage of the k-th synthetic sample, and let $\rho_{k,t}(\theta)$ denote the probability ratio between the current policy and the old policy at the t-step token. The optimization objective is as follows:

$$\mathcal{L}_{\text{GRPO}}(\theta) = -\mathbb{E}_{\{\hat{z_k}\}_{k=1}^{K} \overset{G_\theta}{\sim} z, z\sim\mathcal{D}}\Big[\min\big(\rho_{k,t}(\theta)A_k, \text{clip}\left(\rho_{k,t}(\theta), 1-\epsilon_c, 1+\epsilon_c\right)A_k\big)\Big] \tag{4}$$

This objective increases the generation probability of high-reward outputs, thereby encouraging the generator to produce more realistic, diverse, and structurally valid AM synthetic data.

3.3 Discriminator Refinement Stage

This stage focuses on iteratively improving the discriminator's ability to distinguish annotated AM samples from increasingly high-quality synthetic ones generated by the evolving generator.

Discriminator Initialization. For the discriminator D_ϕ, each training instance is a candidate set consisting of one annotated sample and K synthetic samples. The synthetic samples may contain various types of noise, including unnatural expressions, misaligned argument spans, or invalid relations. Given a shuffled candidate set C, the discriminator outputs a probability distribution over all positions $[p_\phi(j \mid C)]_{j=0}^{K+1}$, where each value is the likelihood that the corresponding sample is the real one. Here, the index of the annotated sample is used as the supervision signal. The equation is as follows:

$$\mathcal{L}_{\text{SFT}}(\phi) = -\log p_\phi(y_{position} \mid C) \tag{5}$$

Based on this, the discriminator is trained as a multi-class classifier to identify the real sample, thereby gradually learning to detect quality issues in textual realization and structural annotations.

Supervised Refinement. After each generator update, the improved generator produces more realistic and challenging synthetic samples. These newly generated samples are then incorporated into the training data to further fine-tune the discriminator using the supervised objective in Eq. 5. As a result, the discriminator progressively becomes more sensitive to subtle structural inconsistencies and annotation errors, leading to stronger discrimination ability against high-quality synthetic samples produced by the evolving generator.

3.4 Adversarial Evolution

Our framework is optimized through iterative adversarial evolution. In the r-th evolution round, the current generator G^r_θ first produces multiple synthetic AM samples conditioned on an annotated

sample. A candidate set is then constructed, and the discriminator identifies the real sample among the candidates. The discriminator output is converted into a reward signal and further combined with the source similarity penalty. The resulting reward is used to optimize the generator via GRPO, producing an updated generator G_θ^{r+1}. These newly generated data are subsequently used to further refine the discriminator D_ϕ^{r+1}, improving its ability to distinguish real samples from synthetic samples. Through this alternating optimization process, the generator and discriminator evolve together progressively. The detailed pseudocode of this process is provided in Appendix B.

4 Experiments

4.1 Datasets

We conduct experiments on three widely used AM datasets: AbstRCT (Mayer et al., 2020), AAEC (Stab and Gurevych, 2017), and CDCP (Park and Cardie, 2018). Each dataset contains argumentative texts and complete manually annotated results. Additional details about the datasets' construction and statistics are provided in Appendix C.1. Following Bao et al., our experiments are performed under two training data settings: using 100% of the original training data and a low-resource setting with only 5% of the training data.

4.2 Implementation Details

In this work, we use Qwen3-8B as the base model for both the generator and the discriminator. We set the number of adversarial evolution rounds to r=3, and use K=3, and λ_1=0.5 in all experiments. Following Bao et al., we limit the scale of the generated synthetic data to twice the size of the original training data used under the selected setting. All experiments were conducted on an NVIDIA H20 GPU. For all experiments[3], we report the average results obtained over five different random seeds.

We use three fine-grained F1 scores from prior work as evaluation metrics: $\mathbf{F1_{span}}$ measures the matching accuracy of component spans, $\mathbf{F1_{aci}}$ evaluates the accuracy of component type classification, and $\mathbf{F1_{ari}}$ assesses the mapping accuracy of argumentative relations.

[3]Code is available at https://github.com/ZhangZJ-GH/Synthetic-Data-for-Argument-Mining.

4.3 AM Models

To examine the effectiveness of synthetic data under different model settings, we adopt three AM models. (1) **ST** (Morio et al., 2022) represents parsing-based AM methods and explicitly performs argument component identification, component type classification, and argumentative relation prediction. (2) **UniASA** (Bao et al., 2025b) is a task-specific generative AM method by reformulating AM as a seq2seq generation task, and we adopt its single-view version to reduce computational cost. (3) **Directly-SFT** serves as an LLM-native generative baseline, where AM data are constructed as instruction-tuning samples and LLMs are directly fine-tuned to generate argument structures. Here, we use Qwen3-8B (Yang et al., 2025) as the base model.

4.4 Compared Methods

We compare our framework with the following approaches: (1) **EDA** (Easy Data Augmentation), which applies synonym replacement to original argumentative texts; (2) **FTGA** (Few-shot Text Generation and Auto-annotation), which uses GPT-4o to generate new texts and then labels them with an AM model; (3) **JTLS** (Joint Text and Label Synthesis), which prompts GPT-4o to generate both texts and AM annotations. (4) **QOS** (Quality-Oriented Synthesis), which rewrites original samples while preserving their argument structures. (5) **DOS** (Diversity-Oriented Synthesis), which generates diverse samples by varying topics and argumentation patterns. (6) **QOS+DOS** (Bao et al., 2025a), which combines 25% QOS data and 75% DOS data, as this combination has been shown to achieve the best performance among different mixing ratios; and (7) **SFTSyn** (Supervised Fine-Tuned Synthesis), where LLMs are fine-tuned to directly generate synthetic text-annotation pairs. Here, we use Qwen3-8B as the LLM. More implementation details are provided in Appendix C.2.

4.5 Main Results

Table 1 reports the main results of our proposed framework and other synthetic data generation methods on the ST, UniASA, and Directly-SFT models. Here, "Origin" denotes training using only the original annotated data. From the results, we draw the following observations.

(1) Incorporating the synthetic data generated by our framework consistently improves the perfor-

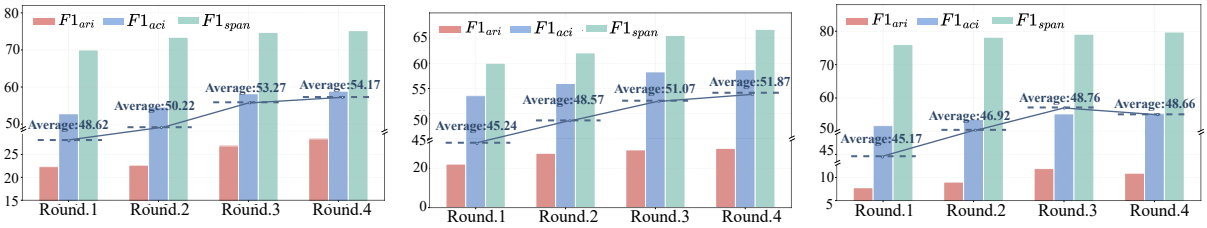
AM Model	Setting	Method	AAEC				AbstRCT				CDCP			
			$F1_{span}$	$F1_{aci}$	$F1_{ari}$	**Avg.**	$F1_{span}$	$F1_{aci}$	$F1_{ari}$	**Avg.**	$F1_{span}$	$F1_{aci}$	$F1_{ari}$	**Avg.**
ST	100%	Origin	84.70	75.88	54.16	71.58	69.01	63.34	38.43	56.93	82.12	68.60	31.07	60.60
		EDA	85.20	76.62	54.18	72.00	69.74	63.91	40.07	57.91	82.37	68.64	30.97	60.66
		FTGA	85.71	77.00	54.60	72.44	70.47	63.98	40.16	58.20	81.89	68.54	32.95	61.13
		JTLS	85.60	76.35	54.56	72.17	70.15	64.32	40.64	58.37	82.22	68.69	30.95	60.62
		QOS	86.19	76.57	56.75	73.17	72.53	66.08	39.49	59.37	82.72	68.62	33.75	61.70
		DOS	85.63	76.94	55.98	72.85	71.90	65.41	38.43	58.58	83.04	67.48	33.17	61.23
		QOS+DOS	85.98	77.08	56.99	73.35	73.19	67.12	40.19	60.17	83.72	69.06	34.66	62.48
		SFTSyn	84.11	75.07	52.33	70.50	68.14	62.03	33.84	54.67	81.00	66.52	29.59	59.04
		Ours	**88.16**	**79.63**	**59.87**	**75.89**	**74.03**	**69.45**	**41.74**	**61.74**	**85.32**	**70.03**	**38.16**	**64.50**
	5%	Origin	68.24	50.42	16.96	45.21	57.17	50.13	21.50	42.93	76.84	49.03	3.40	43.09
		EDA	67.05	48.49	16.48	44.01	58.00	51.16	21.52	43.56	76.54	47.80	3.64	42.66
		FTGA	69.50	52.93	20.88	47.77	56.61	50.30	22.95	43.29	76.40	49.28	4.44	43.37
		JTLS	71.16	52.51	20.23	47.97	59.12	50.32	22.20	43.88	77.61	48.96	3.20	43.26
		QOS	70.84	56.52	23.04	50.13	61.87	54.95	28.21	48.34	77.42	51.55	7.03	45.33
		DOS	71.62	54.85	23.12	49.86	60.58	53.01	23.24	45.61	76.59	52.22	6.93	45.25
		QOS+DOS	72.10	56.02	23.45	50.52	63.04	56.22	28.18	49.15	77.60	54.80	7.87	46.76
		SFTSyn	68.35	48.11	15.55	44.00	57.14	49.93	19.71	42.26	75.01	48.23	4.93	42.72
		Ours	**74.74**	**58.19**	**26.88**	**53.27**	**65.53**	**58.30**	**29.39**	**51.07**	**79.10**	**55.24**	**11.93**	**48.76**
UniASA	100%	Origin	86.72	77.13	56.21	73.35	74.67	66.05	34.01	58.24	81.85	67.08	25.57	58.17
		EDA	87.02	77.69	56.95	73.89	73.64	66.34	34.30	58.09	81.60	67.08	24.58	57.75
		FTGA	87.18	77.68	56.70	73.85	74.90	66.10	35.39	58.80	82.05	67.93	24.73	58.24
		JTLS	87.08	77.62	56.56	73.75	74.98	66.52	35.31	58.94	82.46	67.87	25.08	58.47
		QOS	87.50	79.02	57.58	74.70	75.75	67.33	36.32	59.80	82.05	66.12	25.22	57.80
		DOS	87.49	79.41	57.61	74.84	74.06	66.95	37.67	59.56	77.90	57.89	25.28	53.69
		QOS+DOS	88.15	78.60	58.81	75.19	75.98	68.12	39.41	61.17	82.32	68.04	27.38	59.25
		SFTSyn	85.96	75.68	53.85	71.83	73.11	65.01	31.54	56.55	80.75	65.00	24.17	56.64
		Ours	**89.54**	**80.83**	**61.24**	**77.20**	**77.92**	**68.45**	**42.28**	**62.88**	**84.10**	**69.65**	**30.11**	**61.29**
	5%	Origin	54.50	34.87	5.35	31.57	57.25	44.33	12.69	38.09	71.35	32.03	2.90	35.43
		EDA	61.68	41.42	6.35	36.48	58.03	41.42	12.96	37.47	72.69	35.68	4.67	37.68
		FTGA	69.85	47.81	7.98	41.88	57.87	46.18	15.31	39.79	73.48	34.33	5.31	37.71
		JTLS	58.84	39.09	6.37	34.77	57.60	43.33	16.34	39.09	73.25	37.83	3.56	38.21
		QOS	75.92	57.33	10.15	47.80	59.31	52.00	18.44	43.25	71.60	45.32	6.98	41.30
		DOS	74.45	54.04	8.69	45.73	59.94	50.69	16.59	42.41	71.91	44.19	8.01	41.37
		QOS+DOS	76.69	57.19	13.12	49.00	60.32	54.22	22.80	45.78	72.83	**47.07**	8.19	42.70
		SFTSyn	60.73	42.76	6.23	36.57	56.09	43.78	13.40	37.76	70.44	33.69	2.03	35.39
		Ours	**77.60**	**58.26**	**15.03**	**50.30**	**62.64**	**56.64**	**26.13**	**48.47**	**74.40**	46.72	**11.03**	**44.05**
Directly-SFT	100%	Origin	78.54	66.83	37.64	61.00	73.07	65.52	47.03	61.87	77.32	59.04	21.38	52.58
		EDA	78.92	67.37	37.98	61.42	72.95	65.95	47.82	62.24	77.25	58.86	20.86	52.32
		FTGA	79.28	67.51	38.52	61.77	73.76	65.86	48.37	62.66	77.55	59.18	22.05	52.93
		JTLS	79.16	67.42	38.39	61.66	73.67	66.14	48.65	62.82	77.74	59.12	21.24	52.70
		QOS	79.83	66.63	39.41	61.96	73.49	66.27	49.30	63.02	77.14	56.70	23.40	52.41
		DOS	82.02	67.95	40.09	63.35	73.17	65.41	48.94	62.51	76.66	58.23	23.46	52.78
		QOS+DOS	81.99	70.13	39.51	63.88	74.01	66.74	49.21	63.32	78.15	57.65	24.58	53.46
		SFTSyn	79.03	66.35	34.43	59.94	70.63	65.05	47.41	61.03	74.37	57.83	17.44	49.88
		Ours	**84.48**	**71.10**	**42.91**	**66.16**	**74.51**	**73.88**	**52.80**	**67.06**	**79.66**	**59.57**	**28.03**	**55.75**
	5%	Origin	64.76	51.26	11.03	42.35	64.94	56.82	35.17	52.31	67.92	37.95	9.28	38.38
		EDA	65.12	50.84	10.58	42.18	65.42	56.08	35.31	52.27	68.36	38.44	9.85	38.88
		FTGA	65.86	51.72	12.38	43.32	65.24	57.22	36.92	53.13	68.81	38.76	10.72	39.43
		JTLS	66.28	51.62	12.14	43.35	66.04	56.56	37.14	53.25	69.15	38.62	9.96	39.24
		QOS	65.94	51.26	11.60	42.93	67.23	56.08	37.76	53.69	69.58	35.15	10.54	38.42
		DOS	66.01	51.94	12.89	43.61	66.71	55.30	36.42	52.81	69.03	36.74	11.06	38.94
		QOS+DOS	66.95	51.59	14.12	44.22	67.44	55.32	37.83	53.53	69.60	37.77	11.83	39.73
		SFTSyn	63.94	50.88	6.48	40.43	62.87	56.11	35.58	51.52	63.17	36.37	5.76	35.10
		Ours	**68.48**	**52.91**	**19.07**	**46.82**	**68.92**	**62.47**	**38.78**	**56.72**	**71.97**	**40.13**	**13.62**	**41.91**

Table 1: Main experimental results under full (100%) and low-resource (5%) training data settings. **Bold** numbers denote the best performance among all methods on each dataset. "**Avg.**" is the arithmetic mean of the three F1 scores. The significance test results indicate that all improvements on Avg. are statistically significant with $p < 0.05$.

mance of all three AM models across all datasets and data settings. Compared with training solely on the original data, our framework achieves stable gains under both full-data and low-resource scenarios, indicating that its generated data provides effective complementary training signals. In particular, under the 5%-setting, the ST model achieves average F1 gains of 8.06%, 8.14%, and 5.67% on the three datasets, respectively. This shows that our framework can generate synthetic AM data with

Setting	Method	AAEC				AbstRCT				CDCP			
		$F1_{span}$	$F1_{aci}$	$F1_{ari}$	**Avg.**	$F1_{span}$	$F1_{aci}$	$F1_{ari}$	**Avg.**	$F1_{span}$	$F1_{aci}$	$F1_{ari}$	**Avg.**
100%	**Ours**	**88.16**	**79.63**	**59.87**	**75.89**	**74.03**	**69.45**	**41.74**	**61.74**	**85.32**	**70.03**	**38.16**	**64.50**
	w/o Disc. Refine	84.97	75.75	57.11	72.61	72.23	66.52	40.83	59.86	83.86	67.50	34.30	61.89
	w/o Sim. Penalty	84.44	78.85	59.32	74.20	73.86	68.93	41.25	61.35	84.84	68.85	37.10	63.60
5%	**Ours**	**74.74**	**58.19**	**26.88**	**53.27**	**65.53**	**58.30**	**29.39**	**51.07**	**79.10**	**55.24**	**11.93**	**48.76**
	w/o Disc. Refine	73.02	56.81	25.54	51.79	63.94	55.94	28.91	49.60	77.28	53.27	8.60	46.38
	w/o Sim. Penalty	74.08	56.47	25.96	52.17	63.55	57.61	27.88	49.68	78.82	54.83	10.64	48.10

Table 2: Ablation study. Here, we use the ST model for AM.



(a) On the AAEC dataset. (b) On the AbstRCT dataset. (c) On the CDCP dataset.

Figure 2: Performance comparison of different rounds. Here, we use the ST model for AM.

both high quality and strong diversity.

(2) For PLM-based AM models (ST and Uni-ASA), the improvements brought by our framework are especially evident in the low-resource 5%-setting, suggesting that the proposed framework effectively alleviates the problem of limited annotated data. However, this trend becomes less pronounced for the LLM-based AM model (Directly-SFT). For example, on the AAEC dataset, our framework improves the SFT model by 4.47% under the 5%-setting and 5.16% under the full-data setting, showing only a small difference between the two scenarios. We conjecture that LLM-based AM models are inherently less sensitive to data scarcity due to their stronger prior knowledge.

(3) Our framework generally outperforms other synthetic data generation methods, including the GPT-4o-based QOS+DOS approach. Although these baselines also provide improvements over "Origin", the gains achieved by our framework are more consistent and substantial across different datasets, AM models, and settings. For instance, compared with the QOS+DOS baseline, our framework achieves average F1 improvements of 2.22%, 2.6%, and 1.84% on the three datasets under the 5%-setting. This indicates that the synthetic data generated by our framework is of higher quality and better suited for end-to-end AM tasks.

4.6 Ablation Study

To verify the effectiveness of each component in our framework, we examine two variants: (1) "*w/o Disc. Refine*", which removes discriminator refinement during adversarial training, and (2) "*w/o Sim. Penalty*", which removes the similarity penalty from the final reward. The ablation results are shown in Table 2. We make the following observations. First, removing discriminator refinement consistently degrades performance. This suggests that, without iterative refinement, the discriminator fails to distinguish increasingly challenging synthetic samples generated in later training stages, thereby reducing the quality of the reward signals. Second, removing the similarity penalty also leads to noticeable performance drops. This indicates that the penalty effectively discourages overly similar generations and alleviates the negative impact of repetitive synthetic texts on AM model training.

4.7 Analysis and Discussion

Effect of Evolution Rounds on Performance. We investigate how the rounds of evolution affect performance. Figure 2 shows the results on the three datasets. Overall, as the number of evolution rounds increases, the average performance continues to improve, indicating that adversarial evolution can gradually enhance the quality of synthetic AM data. However, the improvements gradually plateau with more rounds, and slight fluctuations appear on CDCP, likely due to the optimization bottleneck between the generator and discriminator. Considering the increasing training cost of iterative adversarial optimization, we set the number of evolution rounds to $r = 3$ in the main experiments to balance effectiveness and efficiency.

Impact of the Number of Synthetic Samples. We further investigate the effect of the synthetic

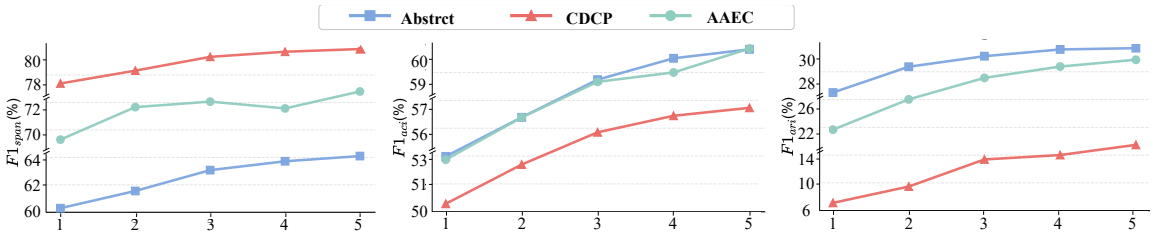
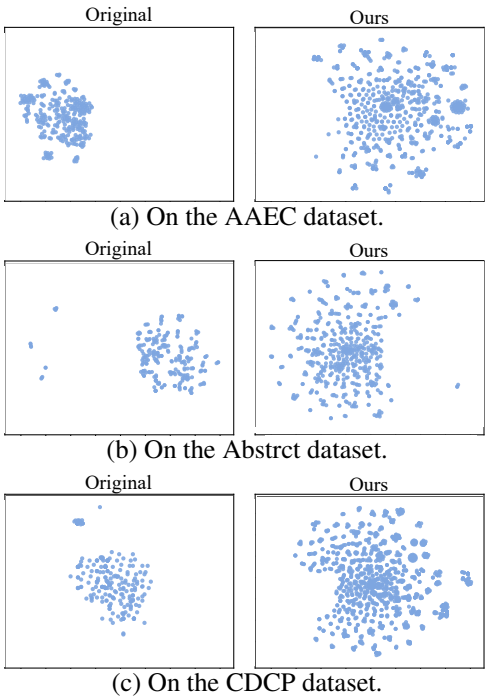


Figure 3: Performance comparison under different numbers of synthetic samples (K) using the ST model for AM.

Setting	Method	AAEC				AbstRCT				CDCP			
		$F1_{span}$	$F1_{aci}$	$F1_{ari}$	**Avg.**	$F1_{span}$	$F1_{aci}$	$F1_{ari}$	**Avg.**	$F1_{span}$	$F1_{aci}$	$F1_{ari}$	**Avg.**
10%	Origin	69.94	52.12	17.53	46.53	58.36	51.65	23.78	44.60	77.95	51.04	5.92	44.97
	Ours	**76.54**	**59.42**	**24.73**	**53.56**	**67.11**	**59.30**	**31.23**	**52.55**	**81.00**	**56.19**	**13.67**	**50.29**
30%	Origin	74.41	60.17	19.04	51.21	62.31	55.78	27.51	48.53	79.36	55.31	12.10	48.92
	Ours	**80.31**	**66.47**	**25.84**	**57.54**	**69.01**	**62.03**	**33.76**	**54.93**	**82.31**	**61.26**	**19.10**	**54.22**
50%	Origin	79.41	67.30	36.48	61.06	65.57	59.61	31.27	52.15	79.76	59.87	15.43	51.69
	Ours	**84.11**	**72.20**	**42.28**	**66.20**	**71.37**	**65.01**	**36.72**	**57.70**	**83.06**	**65.27**	**26.53**	**58.29**
70%	Origin	81.93	73.70	51.49	69.04	67.59	63.19	35.31	55.36	81.56	61.72	20.74	54.67
	Ours	**86.08**	**77.95**	**56.74**	**73.59**	**72.89**	**67.94**	**39.36**	**60.06**	**84.11**	**68.47**	**35.84**	**62.81**

Table 3: Experimental results under different training data settings using the ST model for AM.



(a) On the AAEC dataset.

(b) On the Abstrct dataset.

(c) On the CDCP dataset.

Figure 4: The t-SNE visualization of argument structure distributions under the full-data setting.

sample number K on performance. Figure 3 shows the performance trends under different K values. Overall, all evaluation metrics consistently improve as K increases. This suggests that introducing more synthetic candidate samples provides richer comparison signals for the discriminator, leading to more informative reward feedback for generator optimization. As a result, the generator can produce synthetic data with high-quality and diversity, which further benefits downstream AM models.

Performance under Different Training Data Settings. We further evaluate the performance gains of our framework under four training data settings (10%, 30%, 50%, and 70%). Table 3 presents the relevant experimental results. Consistent with the main experimental results, our framework yields larger improvements in low-resource scenarios, while the gains gradually diminish as more original training data become available. An exception is observed for $F1_{ari}$ on the CDCP dataset, which does not strictly follow this trend. We attribute this to the sparsity of argumentative relation labels in the original CDCP.

Argument Structural Diversity Analysis of Synthetic Data. To evaluate whether our framework improves the structural diversity of synthetic AM data, we follow Bao et al. and visualize the distribution of argument structures. Specifically, we represent argument annotations as graphs, use graph2vec (Narayanan et al., 2017) to obtain graph embeddings, and apply t-SNE for visualization. Notably, this process captures only structural information derived from argument component types and their relations, *without incorporating textual semantic information*. As shown in Figure 4, the synthetic data generated by our framework exhibit a broader and more dispersed structural distribution than the original data across all three datasets. These results indicate that our framework can not only generate

structurally valid AM samples but also produce more diverse argument structures.

Comparison with Other Optimization Methods. To further compare our method with recent synthetic data generation approaches based on different optimization strategies, we select POPri and SEAL as additional baselines, where POPri represents preference-optimization-based data synthesis and SEAL represents reinforcement-learning-based data synthesis (Hou et al., 2025; Zweiger et al., 2026). Since neither method was originally designed for AM data synthesis, we adapt them to the task while preserving their core optimization mechanisms as much as possible, enabling them to generate complete text–structured argument annotation pairs[4]. Table 4 presents the experimental results under the 5% low-resource setting using ST as the AM model. Both POPri and SEAL demonstrate a certain degree of effectiveness across the three datasets, indicating that general preference optimization and reinforcement learning methods can be transferred to AM data synthesis to some extent. However, our method achieves better overall performance than both approaches on all three datasets, further demonstrating the effectiveness of task-specific optimization for AM data synthesis.

Impact of Structure-Aware Reward. To further analyze the impact of explicit structural supervision on data synthesis, we introduce a structure-aware reward on top of the original reward function. For each generated text–annotation pair, we use an AM model trained only on the corresponding original training data and kept frozen as a verifier, and compute the structural reward based on the consistency between its predictions and the generated annotations in terms of argument component boundaries, component types, and argumentative relations, together with basic annotation validity checks. Table 5 presents the corresponding experimental results. We can see that introducing the structure-aware reward further improves the overall performance on AbstRCT and AAEC, indicating that explicit structural supervision can provide additional structural information beyond the existing adversarial feedback. In contrast, the overall performance on CDCP decreases slightly, possibly because the texts in this dataset are shorter and the argument structures are relatively simple, limiting the additional information provided by the structural reward. Overall, these results show that explicit structural supervision can serve as an effective complement to the adversarial reward in our framework, although its effect depends on the characteristics of the specific dataset.

[4]More detailed descriptions of POPri and SEAL, their task adaptation strategies, and the experimental settings are provided in Appendix C.2

Dataset	Method	$F1_{span}$	$F1_{aci}$	$F1_{ari}$	Avg.
AAEC	POPri	72.64	54.91	21.62	49.72
	SEAL	72.73	56.24	20.14	49.70
	SFTSyn	68.35	48.11	15.55	44.00
	Ours	**74.74**	**58.19**	**26.88**	**53.27**
AbstRCT	POPri	62.69	54.82	22.41	46.64
	SEAL	63.84	55.86	20.10	46.60
	SFTSyn	57.14	49.93	19.71	42.26
	Ours	**65.53**	**58.30**	**29.39**	**51.07**
CDCP	POPri	76.88	51.89	9.10	45.96
	SEAL	77.06	52.93	6.13	45.37
	SFTSyn	75.01	48.23	4.93	42.72
	Ours	**79.10**	**55.24**	**11.93**	**48.76**

Table 4: Performance comparison of different training methods using ST as the AM model.

Dataset	Method	$F1_{span}$	$F1_{aci}$	$F1_{ari}$	Avg.
AAEC	Ours	74.74	58.19	26.88	53.27
	w/ Structure Reward	75.24	59.93	27.07	**54.08**
AbstRCT	Ours	65.53	58.30	29.39	51.07
	w/ Structure Reward	66.47	58.93	31.40	**52.27**
CDCP	Ours	79.10	55.24	11.93	**48.76**
	w/ Structure Reward	77.95	54.14	12.64	48.24

Table 5: Effect of the structure-aware reward on downstream AM performance using the ST model for AM.

In addition to the analyses above, we also provide supplementary discussions in the Appendix, including Discourse Diversity Analysis of Synthetic Data, Robustness Analysis under Different Model Configurations, Adversarial Training Stability and Mode Collapse Analysis, Computational Efficiency Analysis and Case Study.

5 Conclusion

This paper proposes an *adversarial reinforcement learning framework for data synthesis*, aimed at alleviating the scarcity of high-quality data in AM. The generator produces text-annotation pairs, while the discriminator provides quality feedback to guide iterative refinement. Experiments on three datasets show that our framework can significantly improve the performance of existing AM models and effectively alleviate the data scarcity problem.

Limitations

Although the proposed framework achieves consistent improvements across multiple experimental

settings, it still has several limitations.

- First, to ensure the quality of the generated data, our method requires multiple rounds of optimization for both the generator and the discriminator, and the adversarial evolution process inevitably introduces additional computational overhead. Future work can further explore more efficient evolution strategies to reduce computational costs while maintaining the quality of synthetic data.
- Second, the effectiveness of our framework depends to some extent on the inherent capabilities of the generator and the discriminator. If the generator itself lacks stable long-text generation capability, the final performance will also be affected.
- Finally, our method still relies on some existing high-quality human-annotated data, and future work needs to explore methods for generating high-quality data from scratch.

Ethics Statement

This paper uses large language models to generate synthetic data for the argument mining task. All experiments are based on publicly available AM datasets, and the generated samples are used only for data augmentation and model training. The chosen datasets are understood to be free of personally identifiable information and offensive content.

Since LLM-generated content may contain biases, factual errors, or invalid annotations, this paper performs format and structural validity checks on the generated samples to filter out low-quality data. However, automatic checking cannot fully guarantee semantic correctness, so human review is still recommended in practical applications or high-risk scenarios. During the writing of this paper, AI tools were used only for grammar checking and language polishing.

Acknowledgments

This research was supported by the Natural Science Foundation for Top Talents of SZTU (Nos. GDRC202518 and GDRC202320).

References

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 ACL-IJCNLP*, pages 6354–6364.

Jianzhu Bao, Yuqi Huang, Yang Sun, Wenya Wang, Yice Zhang, Bojun Jin, and Ruifeng Xu. 2025a. Exploring quality and diversity in synthetic data generation for argument mining. In *Proceedings of EMNLP*, pages 26592–26615.

Jianzhu Bao, Mohan Jing, Kuicai Dong, Aixin Sun, Yang Sun, and Ruifeng Xu. 2025b. Uniasa: A unified generative framework for argument structure analysis. *Computational Linguistics*, 51(3):739–784.

Jianzhu Bao, Rui Wang, Yasheng Wang, Aixin Sun, Yitong Li, Fei Mi, and Ruifeng Xu. 2023. A synthetic data generation framework for grounded dialogues. In *Proceedings of ACL*, pages 10866–10882.

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

Guizhen Chen, Liying Cheng, Luu Anh Tuan, and Lidong Bing. 2024. Exploring the potential of large language models in computational argumentation. In *Proceedings of ACL*, pages 2309–2330.

Oana Cocarascu and Francesca Toni. 2017. Identifying attack and support argumentative relations using deep learning. In *Proceedings of EMNLP*, pages 1374–1379.

Haixing Dai, Zhengliang Liu, Wenxiong Liao, Xiaoke Huang, Yihan Cao, Zihao Wu, Lin Zhao, Shaochen Xu, Fang Zeng, and Wei Liu. 2025. Auggpt: Leveraging chatgpt for text data augmentation. *IEEE Transactions on Big Data*, 11(3):907–918.

Debopam Das and Manfred Stede. 2018. Developing the bangla rst discourse treebank. In *Proceedings of LREC*.

Subhabrata Dutta, Jeevesh Juneja, Dipankar Das, and Tanmoy Chakraborty. 2022. Can unsupervised knowledge transfer from social discussions help argument mining? In *Proceedings of ACL*, pages 7774–7786.

Steffen Eger, Johannes Daxenberger, and Iryna Gurevych. 2017. Neural end-to-end learning for computational argumentation mining. In *Proceedings of ACL*, pages 11–22.

Xu Guo and Yiqiang Chen. 2024. Generative ai for synthetic data generation: Methods, challenges and the future. *arXiv preprint arXiv:2403.04190*.

Alex Havrilla, Andrew Dai, Laura O'Mahony, Koen Oostermeijer, Vera Zisler, Alon Albalak, Fabrizio Milo, Sharath Chandra Raparthy, Kanishk Gandhi, and Baber Abbasi. 2024. Surveying the effects of quality, diversity, and complexity in synthetic data from large language models. *arXiv preprint arXiv:2412.02980*.

Charlie Hou, Mei-Yu Wang, Yige Zhu, Daniel Lazar, and Giulia Fanti. 2025. Private federated learning using preference-optimized synthetic data. In *Proceedings of the 42nd ICML*, pages 24025–24044.

Yohan Jo, Seojin Bang, Chris Reed, and Eduard Hovy. 2021. Classifying argumentative relations using logical mechanisms and argumentation schemes. *TACL*, 9:721–739.

Martin Josifoski, Marija Sakota, Maxime Peyrard, and Robert West. 2023. Exploiting asymmetry for synthetic training data generation: Synthie and the case of information extraction. In *Proceedings of EMNLP*, pages 1555–1574.

Masayuki Kawarada, Tsutomu Hirao, Wataru Uchida, and Masaaki Nagata. 2024. Argument mining as a text-to-text generation task. In *Proceedings of EACL*, pages 2002–2014.

John Lawrence and Chris Reed. 2019. Argument mining: A survey. *Computational Linguistics*, 45(4):765–818.

Marco Lippi and Paolo Torroni. 2015. Context-independent claim detection for argument mining. In *IJCAI*, volume 15, pages 185–191. Buenos Aires.

Tobias Mayer, Elena Cabrio, and Serena Villata. 2020. Transformer-based argument mining for healthcare applications. In *Proceedings of ECAI*.

Marie-Francine Moens, Erik Boiy, Raquel Mochales Palau, and Chris Reed. 2007. Automatic detection of arguments in legal texts. In *Proceedings of ICAIL*, pages 225–230.

Gaku Morio, Hiroaki Ozaki, Terufumi Morishita, and Kohsuke Yanai. 2022. End-to-end argument mining with cross-corpora multi-task learning. *TACL*, 10:639–658.

Annamalai Narayanan, Mahinthan Chandramohan, Rajasekar Venkatesan, Lihui Chen, Yang Liu, and Shantanu Jaiswal. 2017. graph2vec: Learning distributed representations of graphs. *arXiv preprint arXiv:1707.05005*.

Joonsuk Park and Claire Cardie. 2018. A corpus of erulemaking user comments for measuring evaluability of arguments. In *Proceedings of LREC*.

Isaac Persing and Vincent Ng. 2016. End-to-end argumentation mining in student essays. In *Proceedings of NAACL-HLT*, pages 1384–1394.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, and Yang Wu. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

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

Manfred Stede. 2016. Towards assessing depth of argumentation. In *Proceedings of COLING*, pages 3308–3317.

Yang Sun, Guanrong Chen, Caihua Yang, Jianzhu Bao, Bin Liang, Xi Zeng, Min Yang, and Ruifeng Xu. 2024. Discourse structure-aware prefix for generation-based end-to-end argumentation mining. In *Findings of ACL*, pages 11597–11613.

Zhen Tan, Dawei Li, Song Wang, Alimohammad Beigi, Bohan Jiang, Amrita Bhattacharjee, Mansooreh Karami, Jundong Li, Lu Cheng, and Huan Liu. 2024. Large language models for data annotation and synthesis: A survey. In *Proceedings of EMNLP*, pages 930–957.

Henning Wachsmuth, Martin Potthast, Khalid Al Khatib, Yamen Ajjour, Jana Puschmann, Jiani Qu, Jonas Dorsch, Viorel Morari, Janek Bevendorff, and Benno Stein. 2017. Building an argument search engine for the web. In *Proceedings of ArgMining*, pages 49–59.

Chenglong Wang, Yang Gan, Yifu Huo, Yongyu Mu, Qiaozhi He, Murun Yang, Tong Xiao, Chunliang Zhang, Tongran Liu, and Jingbo Zhu. 2024. Lrhp: Learning representations for human preferences via preference pairs. *arXiv preprint arXiv:2410.04503*.

Chenglong Wang, Yang Gan, Hang Zhou, Chi Hu, Yongyu Mu, Kai Song, Murun Yang, Bei Li, Chunliang Zhang, Tongran Liu, and 1 others. 2026a. Mro: Enhancing reasoning in diffusion language models via multi-reward optimization. In *Proceedings of NeurIPS*, pages 29681–29712.

Chenglong Wang, Yifu Huo, Yang Gan, Qiaozhi He, Qi Meng, Bei Li, Yan Wang, Junfu Liu, Tianhua Zhou, Jingbo Zhu, and 1 others. 2026b. Msrl: Scaling generative multimodal reward modeling via multi-stage reinforcement learning. In *Proceedings of CVPR*, pages 29410–29420.

Qianlong Wang, Zhiyuan Wen, Qin Zhao, Min Yang, and Ruifeng Xu. 2021. Progressive self-training with discriminator for aspect term extraction. In *Proceedings of EMNLP*, pages 257–268.

Rui Wang, Jianzhu Bao, Fei Mi, Yi Chen, Hongru Wang, Yasheng Wang, Yitong Li, Lifeng Shang, Kam-Fai Wong, and Ruifeng Xu. 2023. Retrieval-free knowledge injection through multi-document traversal for dialogue models. In *Proceedings of ACL*, pages 6608–6619.

Jason Wei and Kai Zou. 2019. EDA: Easy data augmentation techniques for boosting performance on text classification tasks. In *Proceedings of EMNLP-IJCNLP*, pages 6382–6388, Hong Kong, China. Association for Computational Linguistics.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, and Chenxu Lv. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Yuxiao Ye and Simone Teufel. 2021. End-to-end argument mining as biaffine dependency parsing. In *Proceedings of EACL*, pages 669–678.

Meishan Zhang, Gongyao Jiang, Shuang Liu, Jing Chen, and Min Zhang. 2024. Llm-assisted data augmentation for chinese dialogue-level dependency parsing. *Computational Linguistics*, 50(3):867–891.

Adam Zweiger, Jyo Pari, Han Guo, Yoon Kim, and Pulkit Agrawal. 2026. Self-adapting language models. In *Proceedings of NeurIPS*, pages 74084–74115.

A Task Formulation

Given an input document x, the goal of end-to-end AM is to predict its complete argument structure $y = (Span, Comp, Rela)$, where $Span$ denotes the set of argument component spans, $Comp$ denotes the set of argument components with type labels, and $Rela$ denotes the set of relations between argument components.

Specifically, this task consists of the following three parts:

(1) **Identifies the boundaries of argument components in x**. Each span is denoted as $s_i \in Span$, where $s_i = (b_i, e_i)$. Here, b_i and e_i denote the start and end character positions of the i-th argument component in x, respectively.

(2) **Assigns a corresponding component type to each span**. Each argument component is represented as $c_i = (b_i, e_i, l_i)$, where l_i denotes the type label of the component, such as Claim or Evidence, and the specific label set is determined by the annotation scheme of each dataset.

(3) **Predicts the relations between pairs of argument components**. Each relation is represented as $r_{ij} = (b_i, e_i, b_j, e_j, l_{ij})$, where (b_i, e_i) and (b_j, e_j) denote the spans of the source and target components, respectively, and l_{ij} denotes the relation label between them, such as Support or Attack. Therefore, the overall objective of E2E-AM is to learn a mapping $f(x) \rightarrow (Span, Comp, Rela)$ that converts an input document into a structured argument representation.

B Algorithm

Below is the pseudo-code algorithm for the proposed framework.

C Experimental Setting

C.1 Datasets

We conducted experiments on three widely used AM datasets: **AbstRCT** (Mayer et al., 2020), **AAEC** (Stab and Gurevych, 2017), and **CDCP** (Park and Cardie, 2018). These datasets are derived from three distinct real-world argumentative scenarios and were manually annotated and constructed according to rigorous guidelines, making them high-quality gold-standard corpora in AM. All three datasets are English-language corpora. Table 6 presents the statistics of the three datasets. The details are as follows:

Algorithm 1 Adversarial Evolution Algorithm

Input: Original AM data $\mathcal{D} = \{z_i\}_{i=1}^{N}$; initialization data of generator $\mathcal{D}_G^0$; initialization data of discriminator $\mathcal{D}_D^0$; a base LLM $\mathcal{M}$; number of synthetic samples K; number of evolution rounds T;
Output: Evolved generator G_θ^T

$G^0 \leftarrow \text{SFT}(\mathcal{M}, \mathcal{D}_G^0)$ (Eq. 1)
$D^0 \leftarrow \text{SFT}(\mathcal{M}, \mathcal{D}_D^0)$ (Eq. 5)
for $r = 0, 1, \ldots, T-1$ **do**
  for each original sample $z = (x, y) \in \mathcal{D}$ **do**
    $u \leftarrow \textsc{Serialize}(z)$ //prompt template enclosure
    $\{\hat{z}_k^r\}_{k=1}^{K} \leftarrow G^r(u)$ //synthesize K data
    $C^r \leftarrow \textsc{Shuffle}(\{z, \hat{z}_1^r, \ldots, \hat{z}_K^r\})$
    $y_{\text{position}} \leftarrow \textsc{Position}(z, C^r)$
    $\mathbf{p}^r \leftarrow D^r(C^r)$ (Eq. 5)
    $R_{\text{dis}}^r \leftarrow \min\left(-\log(p_{\phi_r}(y_{\text{position}} \mid C^r) + \epsilon), c\right)$ (Eq. 2) //adversarial reward
    for $k = 1, 2, \ldots, K$ **do**
      $R_{\text{sim}}^{k,r} \leftarrow \textsc{SourcePenalty}(\hat{z}_k^r, z)$ //rule reward
      $R_k^r \leftarrow R_{\text{dis}}^r + \lambda_1 \times R_{\text{sim}}^{k,r}$ (Eq. 3)
    end for
  end for
  $G^{r+1} \leftarrow \text{GRPO}(G^r, \{R_k^r\})$ (Eq. 4)
  $\hat{\mathcal{D}}^{r+1} \leftarrow G^{r+1}(\mathcal{D}, K)$
  $\mathcal{D}_D^{r+1} \leftarrow \textsc{BuildDiscData}(\mathcal{D}, \hat{\mathcal{D}}^{r+1})$
  $D^{r+1} \leftarrow \text{SFT}(D^r, \mathcal{D}_D^{r+1})$ (Eq. 5)
end for
return G^T

AbstRCT An argument mining dataset for the biomedical domain, with data collected from clinical trial abstracts in the PubMed database. It contains 500 abstracts, annotated with 3,279 argument components and 2,060 relations. The component types include MajorClaim, Claim, and Evidence, while the relation types include Support, Attack, and Partial-Attack.

AAEC An English persuasive essay dataset for argument structure analysis, with data collected from students' persuasive essay writing tasks. After manual annotation, it contains 402 complete essays, with 6,089 argument components and 3,832 argumentative relations annotated. The component types include MajorClaim, Claim:For, Claim:Against, and Premise, while the relation types include Support and Attack. Following Bao et al., we uniformly represent Claim:For and Claim:Against as Claim in this work. This is mainly because LLMs are unable to distinguish the subtle differences between the two types of claims, which does not affect the annotation capability of AM models.

Dataset	5% Train	100% Train	Vali.	Test
AAEC	15	289	32	80
AbstRCT	18	350	50	100
CDCP	27	530	58	150

Table 6: Statistics of the datasets. The 5% and 100% settings differ only in the size of the training split, while the validation and test sets remain unchanged.

CDCP An argument evaluability corpus for real-world user comments, with data collected from public comments on debt collection regulations issued by the U.S. Consumer Financial Protection Bureau. This dataset contains 731 user comments, annotated with 4,779 argument components and 1,353 relations. The component types include Value, Fact, Policy, Testimony, and Reference, while the relation types include Reason and Evidence.

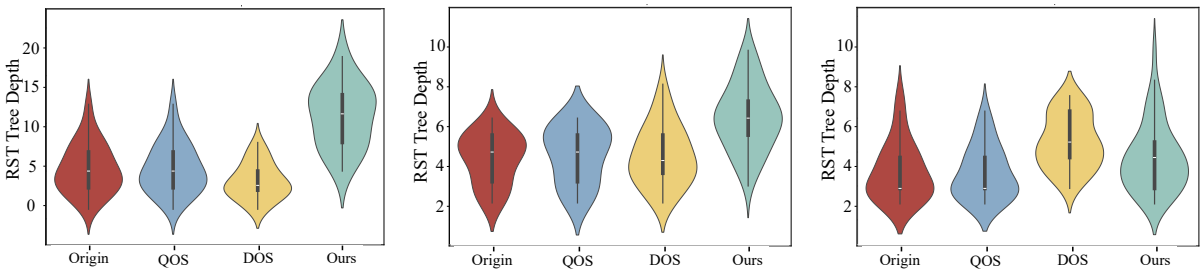
C.2 Compared Method

We compare our data synthesis framework with the following data augmentation and synthetic data generation methods:

- **Easy Data Augmentation** (EDA). This method only performs lexical-level operations on argumentative texts, while the original EDA framework includes lexical operations such as random insertion, deletion, replacement, and synonym replacement (Wei and Zou, 2019). However, since the first three operations may affect the argument structure of the original data, we only apply synonym replacement to the argumentative texts.
- **Few-shot Text Generation and Auto-annotation** (FTGA). FTGA first prompts GPT-4o with several original argumentative texts and asks it to generate new texts following the same domain and writing style. The LLM is not required to generate AM annotations in this setting. Instead, use the AM model trained on the original training set to automatically annotate the generated texts.
- **Joint Text and Label Synthesis** (JTLS). JTLS provides GPT-4o with a few text-annotation examples and requires it to generate both the argumentative text and the corresponding AM annotations.
- **Quality-Oriented Synthesis** (QOS) (Bao et al., 2025a).QOS aims to produce high-quality synthetic samples by preserving the

Setting	AAEC				AbstRCT				CDCP			
	$F1_{span}$	$F1_{aci}$	$F1_{ari}$	**Avg.**	$F1_{span}$	$F1_{aci}$	$F1_{ari}$	**Avg.**	$F1_{span}$	$F1_{aci}$	$F1_{ari}$	**Avg.**
QOS+DOS	72.10	56.02	23.45	50.52	63.04	56.22	28.18	49.15	77.61	54.80	7.87	46.76
Qwen3-1.7B+Qwen3-8B	70.78	53.64	21.05	48.49	59.26	54.53	22.47	45.42	75.16	51.62	4.73	43.84
Qwen3-4B+Qwen3-8B	73.16	56.73	25.37	51.75	63.31	56.80	26.27	48.79	77.79	53.11	7.90	46.27
Qwen3-8B+Qwen3-1.7B	72.54	56.74	24.96	51.41	63.55	56.64	27.11	49.10	78.05	53.86	6.82	46.24
Qwen3-8B+Qwen3-4B	73.89	58.01	26.13	52.68	64.99	57.75	**29.45**	50.73	78.57	54.21	8.23	47.00
Qwen3-8B+Longformer	71.21	54.74	23.42	49.79	62.32	55.62	26.57	48.17	77.48	52.87	6.01	45.45
Qwen3-8B+Qwen3-8B	**74.74**	**58.19**	**26.88**	**53.27**	**65.53**	**58.30**	29.39	**51.07**	**79.10**	**55.24**	**11.93**	**48.76**

Table 7: Robustness analysis under different generator–discriminator model combinations using ST as the AM model. Here, $\mathcal{M}_a$+$\mathcal{M}_b$ denotes using model $\mathcal{M}_a$ as the generator and model $\mathcal{M}_b$ as the discriminator. Longformer refers to Longformer-4096. In the main experiments, both the generator and discriminator are based on Qwen3-8B.



(a) On the AAEC dataset. (b) On the AbstRCT dataset. (c) On the CDCP dataset.

Figure 5: RST tree depth distributions of different argument data under the full-data setting.

original argument structure. Specifically, it remains aligned with the original component and relation annotations, and only performs structure-aware paraphrasing on context and component. In this way, QOS emphasizes annotation reliability and structural consistency.

- **Diversity-Oriented Synthesis** (DOS) (Bao et al., 2025a).DOS focuses on increasing the diversity of synthetic AM data. Instead of mainly rewriting the original instance, it encourages the generation of new samples with different topics, expressions, and argumentation patterns.

- **Supervised Fine-Tuned Synthesis** (SFTSyn). SFTSyn is an additional LLM-based synthesis baseline. We construct paired training instances where the input is an original labeled AM sample and the output is a corresponding synthetic labeled sample. LLM is then supervised and fine-tuned to learn this mapping and directly generate new text-annotation pairs.

- **Policy Optimization for Private Data** (POPri) (Hou et al., 2025). POPri is a preference optimization-based synthetic data generation method that scores and ranks multiple candidate samples to construct preference pairs and further optimizes the generator through DPO. To adapt it to AM data synthesis, we treat the complete text–annotation pair as the generation target and construct preference pairs based on candidate scores computed against the original AM training data. The privacy-related components in the original framework are omitted since they are not applicable to our setting.

- **Self-Adapting LLMs** (SEAL) (Zweiger et al., 2026). SEAL is a reinforcement learning-based method that optimizes synthetic fine-tuning data according to their utility for subsequent model updates and downstream performance. We adapt its self-edit generation process to produce complete AM text–annotation pairs. To reduce the computational cost of repeatedly performing temporary model updates, we use a lightweight proxy reward based on structural validity, similarity, and novelty to select self-edits for policy refinement.

For a fair comparison, POPri and SEAL use Qwen3-8B and start from the same generator SFT initialization as our method. They follow the same data splits and synthetic-data budget.

D Additional Analysis

D.1 Robustness Analysis under Different Model Configurations

In the above experiments, both the generator and discriminator are based on Qwen3-8B. To further evaluate the robustness of our framework, we investigate the effect of different generator–discriminator model-size combinations. Table 7 presents the results using ST as the AM model.

The results show that the final performance is more sensitive to the scale of the generator than that of the discriminator. Specifically, performance drops noticeably under the Qwen3-1.7B+Qwen3-8B setting. In contrast, when the generator remains unchanged, replacing the discriminator with a smaller model leads to only marginal degradation. This indicates that the generator plays a more important role in determining the quality of synthetic AM data. Moreover, even under the Qwen3-4B+Qwen3-8B setting, our framework still achieves performance comparable to or better than the SOTA QOS+DOS baseline on most metrics, demonstrating the robustness of our framework across different model-size configurations.

In addition, we further examine the performance when using a lighter-weight model as the discriminator. Specifically, we use Longformer-Base-4096 as the discriminator while keeping the generator and all other experimental settings unchanged (Beltagy et al., 2020). We choose Longformer rather than BERT-base because it can handle longer input sequences and is therefore better suited for discriminating complete AM text–annotation candidate sets. As shown in Table 7, using Longformer as the discriminator leads to a decline in overall performance across all three datasets. This indicates that, compared with a lighter-weight encoder model, an LLM-based discriminator can provide more effective feedback signals to the generator.

D.2 Discourse Diversity Analysis of Synthetic Data

To further analyze the diversity of the synthetic data generated by our framework at the discourse-structure level, we conduct a discourse structural analysis of argumentative content from different data sources. Specifically, we parse the data into Rhetorical Structure Theory (RST) trees and compare the distributions of their RST tree depths (Stede, 2016). Unlike the previous Argument Structural Diversity Analysis of Synthetic Data, which does not introduce textual semantic information and constructs structural representations only from argument component types and their relations, this analysis directly operates on the generated texts themselves and characterizes the structural richness of synthetic data in terms of discourse organization and rhetorical relations.

As shown in Figure 5, the data generated by our framework exhibit deeper RST tree structures across all three datasets. This indicates that our framework can generate synthetic texts with richer discourse organization structures, rather than merely producing shallow or linear argumentative expressions. Notably, the RST tree-depth distribution of QOS is relatively close to that of the original data. This is because QOS mainly rewrites content while preserving the overall structural arrangement of the original samples, so its generated texts often retain discourse organization patterns similar to those of the original data.

D.3 Adversarial Training Stability and Mode Collapse Analysis

To further analyze the training stability during adversarial evolution, we record the training dynamics of the Generator and Discriminator at different evolution stages, as shown in Table 8. For the Generator, the GRPO reward shows an overall mild upward trend, with only slight fluctuations on AAEC. Meanwhile, the reward standard deviation remains non-zero throughout training, indicating that informative reward differences are preserved among candidate samples. The gradient norm remains relatively small and stable without obvious abnormal fluctuations, while policy entropy decreases gradually rather than collapsing rapidly. Overall, these results indicate that the Generator optimization process is relatively stable, with no obvious reward collapse, gradient explosion, or premature policy collapse observed.

For the Discriminator, we evaluated its prediction accuracy before and after each incremental refinement round. We observe a consistent alternating adaptation pattern: after each Generator update, the Discriminator accuracy temporarily decreases because the newly generated synthetic samples become more challenging, but it consistently recovers and further improves after the subsequent refinement stage. This behavior demonstrates that the Generator continuously produces harder synthetic examples while the Discriminator success-

Dataset	Stage	GRPO Reward	Reward Std.	Grad. Norm	Entropy	D Acc. (Beg.)	D Acc. (End.)
AbstRCT	SFT	–	–	–	–	25.0%	69.1%
	Round 1	1.72 ± 0.057	0.35 ± 0.095	0.08 ± 0.044	0.26 ± 0.008	55.0%	80.0%
	Round 2	1.79 ± 0.034	0.25 ± 0.101	0.07 ± 0.007	0.25 ± 0.007	63.0%	86.2%
	Round 3	1.82 ± 0.032	0.23 ± 0.076	0.07 ± 0.020	0.24 ± 0.016	67.0%	88.4%
CDCP	SFT	–	–	–	–	25.0%	76.4%
	Round 1	1.87 ± 0.026	0.13 ± 0.017	0.10 ± 0.013	0.26 ± 0.008	48.4%	81.2%
	Round 2	1.88 ± 0.029	0.13 ± 0.008	0.12 ± 0.032	0.24 ± 0.007	58.1%	83.7%
	Round 3	1.91 ± 0.028	0.14 ± 0.014	0.11 ± 0.055	0.22 ± 0.008	66.3%	85.6%
AAEC	SFT	–	–	–	–	29.1%	82.0%
	Round 1	1.72 ± 0.057	0.38 ± 0.080	0.07 ± 0.003	0.27 ± 0.002	50.0%	71.5%
	Round 2	1.68 ± 0.009	0.38 ± 0.065	0.06 ± 0.024	0.25 ± 0.004	60.2%	90.0%
	Round 3	1.69 ± 0.020	0.37 ± 0.060	0.06 ± 0.020	0.24 ± 0.006	64.3%	79.5%

Table 8: Training dynamics of the generator and discriminator across adversarial evolution stages. GRPO Reward and Reward Std. respectively denote the reward obtained during generator optimization and the standard deviation of rewards. Grad. Norm and Entropy denote the gradient norm and policy entropy of the generator, respectively. D Acc. (Beg.) and D Acc. (End.) denote the discriminator accuracy at the beginning and end of each refinement stage. SFT denotes the discriminator initialization stage before adversarial evolution.

Dataset	Methods	Time	Memory	Avg.
AbstRCT	Ours	9.3h	60.3GB	**51.07**
	w/ Longformer	8.6h	51.1GB	48.17
	SFTSyn	0.9h	43.4GB	42.26
CDCP	Ours	8.1h	58.2GB	**48.76**
	w/ Longformer	7.9h	47.7GB	45.45
	SFTSyn	1.1h	49.8GB	42.72
AAEC	Ours	7.6h	64.1GB	**53.27**
	w/ Longformer	7.4h	54.7GB	49.79
	SFTSyn	0.8h	46.7GB	44.00

Table 9: Computational cost and downstream AM performance of different synthesis methods. Time denotes the total training time in hours, Memory denotes the peak GPU memory usage in GB, and Avg. denotes the average of the three downstream AM F1 scores. We use ST as the AM model.

fully adapts to the evolving data distribution, leading to stable adversarial co-evolution rather than unstable competition.

This phenomenon also provides a reference for analyzing potential mode collapse. In AM data synthesis, mode collapse manifests as the Generator gradually concentrating on a small number of repetitive or highly similar argumentation patterns. If obvious mode collapse occurred, the Discriminator should be able to identify these repetitive patterns more easily, rather than repeatedly exhibiting accuracy drops after Generator updates. The current results instead show that newly generated samples in each round can continuously pose new challenges to the existing Discriminator. In addition, discriminator feedback encourages the Generator to produce samples that are more difficult to distinguish, while the similarity penalty further suppresses excessive reuse of the original samples. Therefore, within the adversarial evolution rounds examined, we do not observe obvious signs of mode collapse.

D.4 Computational Efficiency Analysis

To assess the computational cost of our method, we compare different approaches in terms of training time, peak GPU memory usage, and downstream AM performance, as shown in Table 9. Compared with the variant using Longformer as the discriminator, our method introduces additional computational and GPU memory overhead, while achieving better downstream performance on all three datasets. In contrast, SFTSyn requires substantially less training time and computational resources, whereas our method additionally involves candidate sampling, GRPO optimization, and iterative discriminator updates, resulting in higher computational costs. However, these additional costs are accompanied by more substantial improvements in AM performance. It should be noted that the generated synthetic data can be reused to train different AM models without introducing additional overhead during downstream inference.

D.5 Case Study

To further intuitively demonstrate the effectiveness of our framework, we select an AM sample for case analysis. We compare the original sample with the generated synthetic sample and examine both the generated text and its structured annotations. Specifically, we focus on 1) whether the text is natural and fluent, 2) whether the argument component spans can accurately match the input text,

3) whether the component labels are correct, and 4) whether the argumentative relations are consistent with the textual logic and semantics.

As shown in Figure 6, we require the model to generate a synthetic sample that is consistent with the topic discussed in the original sample. Even though some data and indicators from the original sample are used, the synthetic sample still exhibits clear differences in its specific content organization. Consequently, the annotatable argument components and logical relations also change accordingly.

E Prompt

For clarity and reproducibility, we provide the prompt used in each step of our framework.

Specifically, the prompt for generator initialization is shown in Figure 7. And the prompt for discriminator initialization is shown in Figure 8.

Original Training Data Instance:

Argumentative Text:

Patients experience reductions in quality of life (QOL) while receiving cancer treatment and several approaches have been proposed to address QOL issues. In this project, the QOL differences between older adult (age 65+) and younger adult (age 18-64) advanced cancer patients in response to a multidisciplinary intervention designed to improve QOL were examined. This study was registered on ClinicalTrials.gov. Those in the intervention group received six multidisciplinary 90-minute sessions designed to address the five major domains of QOL. Outcomes measured at baseline and weeks 4, 27, and 52 included QOL (Linear Analogue Self-Assessment (LASA), Functional Assessment of Cancer Therapy-General (FACT-G)) and mood (Profile of Mood States (POMS)). Kruskall-Wallis methodology was used to compare scores between older and younger adult patients randomized to the intervention. Of 131 patients in the larger randomized controlled study, we report data on 54 evaluable patients (16 older adults and 38 younger adults) randomized to the intervention. Long-term benefits for older patients were seen in the anger-hostility scale at week 27 (92.2 vs. 84.2, p = 0.027) and week 52 (96.3 vs. 85.9, p = 0.005). Older adult patients who received a multidisciplinary intervention to improve QOL while undergoing advanced cancer treatments benefited differently in some QOL domains, compared to younger adult patients. Future studies can provide further insight on how to tailor QOL interventions for these age groups.

Argumentative Component:

< " id ": 0 , " label ": " Evidence ", " Span ": [{" from ": 1185, " to ": 1405}] >

< " id ": 1 , " label ": " Evidence ", " Span ": [{" from ": 1406, " to ": 1560}] >

< " id ": 2 , " label ": " Claim ", " Span ": [{" from ": 1561, " to ": 1765}] >

Argumentative Relations:

< " Source ": 2 , " Target ": 0, " label ": "Support" >

< " Source ": 2 , " Target ": 1, " label ": "Support" >

Synthetic Data Instance:

Argumentative Text:

Recent research suggests that multidisciplinary interventions significantly enhance quality of life (QOL) for older adults with advanced cancer, whereas younger patients show less improvement. A recent randomized controlled trial evaluated the efficacy of a tailored intervention program for QOL in two age groups. The study enrolled 120 patients, with 60 older adults (age ≥65) and 60 younger adults (age 18-64) diagnosed with advanced cancer. Participants were randomly assigned to either a multidisciplinary intervention group or a standard care group. The intervention included weekly counseling sessions, nutritional support, and physical therapy over a 12-week period. QOL was assessed using the FACT-G and LASA scales at baseline, 6 weeks, and 12 weeks. Results indicated that older adults in the intervention group reported a 25% improvement in overall QOL ($p < 0.01$), compared to only a 10% improvement in younger adults ($p = 0.02$). Additionally, older patients demonstrated higher satisfaction with social well-being (FACT-G scores: 88.5 vs. 80.3, $p = 0.03$) and lower levels of anger-hostility (POMS scores: 90.0 vs. 85.5, $p = 0.01$). However, the study also found that younger adults experienced greater improvements in physical function ($p = 0.008$) and emotional well-being ($p = 0.005$). These findings suggest that while multidisciplinary interventions benefit both age groups, older adults may derive more substantial QOL gains. Further research is needed to explore how age-specific factors influence the effectiveness of QOL interventions.

Argumentative Component:

< " id ": 0 , " label ": " Claim ", " Span ": [{" from ": 32, " to ": 192}] >

< " id ": 1 , " label ": " Evidence ", " Span ": [{" from ": 315, " to ": 444}] >

< " id ": 2 , " label ": " Evidence ", " Span ": [{" from ": 556, " to ": 764}] >

< " id ": 3 , " label ": " Evidence ", " Span ": [{" from ": 942, " to ": 1143}] >

Argumentative Relations:

< " Source ": 2 , " Target ": 0, " label ": "Support" >

< " Source ": 3 , " Target ": 0, " label ": "Support" >

Figure 6: Examples of original data and synthetic data.

Prompt for Synthetic Data Generation

You are an expert synthetic data generator for end-to-end argument mining.

You will be given one original annotated argument-mining sample in normalized JSON format.

The sample contains an input text and its corresponding structured argument annotation, including the argument-mining framework, argumentative components, argumentative relations.

Your task is to generate one new synthetic argument-mining sample for data augmentation.

The generated sample should follow the same annotation schema and labeling convention as the original sample, but it should not be a direct copy or a shallow paraphrase of the original sample.
It should introduce a new textual realization and, when appropriate, a different argumentative organization while remaining consistent with the target framework.

Generation requirements:

1. Output format:
- Output exactly one valid JSON object.
2. Framework consistency:
- The field "framework" must remain "{framework}".
- Use only the allowed node labels: {node_labels}.
- Use only the allowed edge labels: {edge_labels}.
3. Text quality:
- The generated input text should be natural, coherent, and semantically complete.
- The text should be suitable for argument mining and should contain explicit argumentative content.
- Avoid directly copying phrases, sentences, or argumentative patterns from the original sample.
4. Argument component validity:
- Each node must represent a meaningful argumentative component in the generated input.
- Each node must use the following format:
{"id": <integer>, "label": "<node_label>", "text": "<exact span from the generated input>"}
- Node ids must be consecutive integers starting from 0.
- Every node["text"] must appear exactly and contiguously in the generated input.
- Do not hallucinate node texts that are not present in the generated input.
5. Label-semantics consistency:
- The label of each node must match the argumentative function of its text span.
- Claims, premises, evidence, major claims, or other framework-specific component types should be assigned according to their actual semantic roles in the generated text.
6. Relation validity:
- Each edge must connect two existing node ids.
- Each edge must use the following format:
{"source": <node_id>, "target": <node_id>, "label": "<edge_label>"}
- The direction and label of each relation must be logically consistent with the meaning of the connected components.
7. Novelty and diversity:
- The generated sample should differ from the original sample in wording and content.
- It should preserve the annotation style of the original data while contributing a new and useful training instance.

Original AM sample:
{reference_sample}

Expected output schema:
{output_schema}

Now generate one new synthetic argument-mining sample as a single JSON object.

Figure 7: Prompt for Synthetic Data Generation.

Prompt for Discriminator SFT Training

You are an expert discriminator for end-to-end argument mining data quality assessment.

You are given four candidate argument-mining samples in normalized JSON format.

Exactly one candidate is an original high-quality labeled sample, while the other three are synthetic samples generated for data augmentation.

Your task is to identify which candidate is most likely to be the original sample.

You must evaluate each candidate as a complete text-annotation pair, rather than judging only the surface fluency of the input text.

Evaluation criteria:

1. Textual quality: the input should be natural, coherent, and semantically complete.
2. Schema validity: the sample should contain valid fields, including input, framework, nodes and edges.
3. Span grounding: every node text should correspond to an exact span in the input text.
4. Component semantics: each node label should match the argumentative role of its span.
5. Relation consistency: each edge should connect valid nodes and express a plausible argumentative relation according to its label.
6. Global structure: the overall argument graph, including tops, should be logically coherent and consistent with the text.

Synthetic samples may appear fluent but can contain subtle defects, such as copied or generic expressions, unsupported claims, mismatched node labels, invalid span boundaries, incorrect relation directions, dangling edges, or implausible argument structures.
Do not rely on superficial cues such as length, formatting, or topic alone.

Choose the candidate that best satisfies both textual naturalness and structural annotation consistency.

Candidata A:
{candidate_0}

Candidata B:
{candidata_1}

...

Candidata K+1:
{candidata_*K*}

Output label semantics:

0 → Candidate A is the original sample

1 → Candidate B is the original sample

...

K → Candidate K is the original sample

Figure 8: Prompt for Discriminator SFT Training.