Argument Component Segmentation with Fine-Tuned Large Language Models

Ettore Caputo, Sergio Greco, Lucio La Cava
DIMES Dept., University of Calabria
v. P. Bucci 44Z, 87036 Rende, CS, Italy
{ettore.caputo, s.greco, lucio.lacava}@dimes.unical.it

Abstract

Argument Mining (AM) aims to identify and interpret argumentative structures in unstructured text, with Argument Component Classification (ACC) as a core task. Despite significant advances, most ACC approaches rely on manually pre-segmented inputs, an assumption that rarely holds in practice due to the high cost and effort of expert human annotation, creating a major bottleneck for scalable AM systems.

In this work, we focus on the foundation Argument Component Segmentation (ACS) task by proposing a fine-grained, paired-tag annotation schema that explicitly distinguishes between relevant and surrounding content, thus overcoming the limitations of previous single-separator approaches. Leveraging small and open Large Language Models (LLMs) fine-tuned on our paired-tag annotation schema, we can perform ACS with quality comparable to human expert annotators across multiple benchmark datasets. We further validate our approach on the downstream ACC task, showing that automated segmentation with fine-tuned LLMs yields ACC performances comparable to pipelines relying on human annotations.

These findings suggest that reliable automated ACS via LLMs is both feasible and effective, paving the way for more scalable AM pipelines without human intervention.

1 Introduction

Argument Mining (AM) has emerged as a key branch of Natural Language Processing dedicated to the automatic detection, segmentation, and interpretation of argumentative discourses in unstructured texts (Lawrence and Reed, 2019). By extracting argument components (e.g., premises and claims) and the relations that bind them, Argument Mining facilitates downstream applications in various fields, including legal analysis (Palau and Moens, 2009), scientific debate (Sukpanichnant et al., 2024), and web discourse (Habernal and Gurevych, 2017).

Argument Mining pipelines typically consists of four key subtasks (Eger et al., 2017; Cabrio and Villata, 2018): (i) *Argument Component Segmentation* (ACS), to distinguish argumentative units from non-argumentative ones; (ii) *Argument Component Classification* (ACC), to assign a specific type to each identified argument component (e.g., claim, premise); (iii) *Argument Relation Identification* (ARI), to detect relations among argument components; and (iv) *Argument Relation Classification* (ARC) to label the detected relations (e.g., attacks, supports).

Traditionally, the majority of research has focused on ACC and ARC tasks, achieving nowadays remarkable results (Liu et al., 2023; Cabessa et al., 2025; Gorur et al., 2025). However, this often presuppose the availability of already identified argument components (Niculae et al., 2017; Liu et al., 2023), contrasting with real-world cases, where texts lack or have incomplete pre-annotated arguments (Peldszus, 2014), and manually annotating these at scale is prohibitively costly and time-consuming (Reed et al., 2008; Lawrence and Reed, 2019). Consequently, this mismatch between research assumptions and practical constraints leaves ACS underexplored, creating a bottleneck for more automated AM systems.

Large Language Models exhibit remarkable Natural Language Understanding capabilities (Chang et al., 2024) due to their ability to capture rich contextual representations and model long-range dependencies in unstructured texts, and represent a promising direction for automated ACS, with reliable boundary detection and distinction between argumentative and non-argumentative pieces of text.

Contributions. Motivated by these considerations, this work addresses a key limitation in current Argument Mining research: the lack of effective and

Findings of the Association for Computational Linguistics: EACL 2026, pages 5154–5167

fine-grained automated segmentation of argument components in unstructured texts. Specifically, our contributions are as follows:

- We propose a *paired*-tag annotation schema that enables a more *fine-grained* identification of argument components. This addresses key limitations of previous single-separator segmentation methods (Favero et al., 2025), which implicitly assume that all text spans belong to some argument component, thus potentially not distinguishing argumentative from non-argumentative content.
- We show that fine-tuned, compact, and openly available LLMs can overcome existing ACS approaches and achieve segmentation quality comparable to human annotators across diverse, human-annotated datasets. By prioritizing efficiency and accessibility, our approach enables scalable deployment and supports the automatic creation of high-quality annotated argumentative corpora with minimal to no human intervention, addressing a key issue in the current landscape (Kashefi et al., 2023).
- We validate our segmentation approach in the downstream ACC task on widely recognized benchmark datasets, showing that our fully automated segmentation leads to classification performances matching the ones based on gold-standard human annotations, detaching from earlier works where single-separator automated segmentation showed disruptive effects on the downstream ACC tasks (Morio et al., 2022; Favero et al., 2025). This highlights the reliability of our method and its practical viability in end-to-end argumentative analysis pipelines, without the need for costly or impractical manual annotation.

The remainder of the paper is organized as follows: Section 2 discusses related work on Argument Mining, Section 3 formalizes the ACS and ACC problems, Section 4 outlines our methodology, Section 5 outlines our evaluation approach, Section 6 presents our results, and Section 7 concludes this work by outlining future directions.

2 Related Work

Argument Mining encompasses various subtasks, including component detection, classification, relation identification, and quality assessment (Cabrio and Villata, 2018). Despite its relevance across domains such as legal analysis (Carstens and Toni, 2015), education (Zhang and Litman, 2016), and scientific discourse (Kirschner et al., 2015), only a limited number of studies have addressed AM in its full complexity, due to the inherent methodological challenges (Chen et al., 2022).

Initial AM approaches relied on traditional supervised machine learning algorithms such as maximum entropy classifiers (Palau and Moens, 2011), logistic regressors (Levy et al., 2014), Support Vector Machines (Stab and Gurevych, 2014; Niculae et al., 2017), optimization techniques (Stab and Gurevych, 2017), or deep neural networks like RNNs (Eger et al., 2017; Niculae et al., 2017) and LSTMs (Potash et al., 2017).

Advances in Transformer-based architectures such as BERT (Mayer et al., 2020; Kashefi et al., 2023), Longformer (Ding et al., 2022), and T5 (Kawarada et al., 2024), have significantly improved performance on classification and relation prediction, due to the greater contextual awareness and ability to catch the argumentative flow in texts.

Recently, generative Large Language Models demonstrated remarkable capabilities in AM tasks framed as text-generation (Chen et al., 2024). In this direction, Liu et al. (2023) used a BART-based text-generation model to learn the argument structure as a Chain-of-Thought, Gorur et al. (2025) employed various open-source and commercial LLMs to perform Relation-based Argument Mining, Pojoni et al. (2023) leveraged ChatGPT to extract arguments from podcast transcripts, whereas Favero et al. (2025) relied on small LLMs for argument mining in educational contexts. Recent works also investigate the effects of different LLM optimizations techniques for AM, such as in-context learning and fine-tuning approaches (Cabessa et al., 2024, 2025). However, to the best of our knowledge, no works assessing the feasibility and impact of fully automated ACS via generative LLMs have been proposed, thus prompting us to fill this gap.

3 Problem Definition

In this section, we formally define two key argument mining sub-tasks, namely the *Argument Component Segmentation* (**ACS**) and *Argument Component Classification* (**ACC**).

Let $D = [t_1, t_2, ..., t_n]$ represent an open-ended textual document as a sequence of tokens, where each $t_i \in V$ is a *word* drawn from a vocabulary V.

Definition 1 (Argument Component) An *argument component* (**AC**) of D is any subsequence $[t_b, ..., t_e]$ of D which has a proper meaning and can be classified according to a given classification schema (e.g., *Claim*, *Premise*).

Definition 2 (Argument Component Segmentation) Let $D = [t_1, t_2, ..., t_n]$, the argument component segmentation (**ACS**) problem consists of finding a mapping function f that applied to D gives in output a list of components $[AC_1, ...AC_m]$, such that $e_i < b_j, \forall i < j \leq m$.

Definition 3 (Tagged Argument Component) Given a document $D = [t_1, t_2, ..., t_n]$ and a mapping function f, and let $f(D) = [AC_1, ...AC_m]$, with $AC_i = [t_{b_i}, ..., t_{e_i}]$, the tagged argument component AC_i is defined as $AC_i^* = [\langle AC_i \rangle, t_{b_i}, ..., t_{e_i}, \langle /AC_i \rangle]$. Accordingly, D^* denotes the tagged document derived from D by replacing each AC_i $(1 \leq i \leq m)$ with the corresponding AC_i^*.

Definition 4 (Argument Component Classification) Given a tagged document D^* and a component index i, the argument component classification (**ACC**) problem consists in defining a function γ assigning a class (e.g., *Premise*, *Claim*, *Major Claim*) to the i-th argument component in D^*.

4 Methodology

Our approach follows a paired-tag annotation schema, as outlined in Definition 3, to achieve fine-grained and more suitable ACS (Stab and Gurevych, 2017; Mayer et al., 2020), contrasting with previous work in Argument Mining (Favero et al., 2025), which relies on sentence-level segmentation. An example of this difference is depicted in Figure 1.

Furthermore, in contrast with Favero et al. (2025), we do not use any additional LLMs to assess argumentative content, yet we propose a unified fine-tuning strategy allowing any single LLM architecture to be optimized for both Argument Component Segmentation (ACS) and Argument Component Classification (ACC).

To foster flexibility, we introduce two task-specific modules: the *Argument Component Identification Module* (ACIM) and the *Argument Component Classification Module* (ACCM). These modules can be fine-tuned independently, depending on the downstream task (e.g., when annotations are already available or classification is not required)

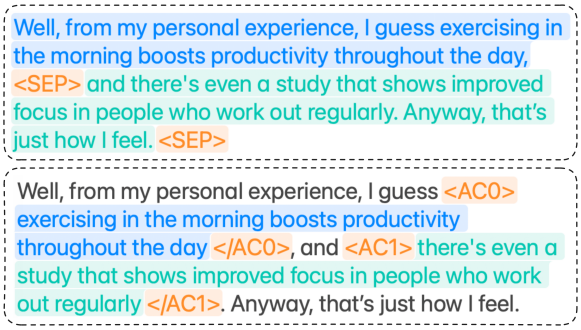

Figure 1: Single (top) vs. paired (bottom) tagging approach for the ACS task. Colors denote identified argument components. Crossed words refer to unnecessary parts included by traditional approaches yet filtered out by our paired-tagging approach.

and leveraged to develop a complete segmentation and classification pipeline. An example of our proposed approach is shown in Figure 2, and we next describe the methodology in detail.

4.1 Learning Paradigm

Due to the inherent complexity of natural language, finding the exact transformation function f and the classification function γ poses significant challenges. Therefore, in this work, we approximate these functions as a *text-generation task* by means of a *decoder-only* Large Language Model whose parameters θ (being them for ACS or ACC) are optimized via fine-tuning.

Our transformation function f is implemented as an LLM model fine-tuned on a paired dataset $(D_i, D_i^\tau)_{i=1}^L$ containing L documents D_i and corresponding ground-truth tagged versions D_i^τ.

Following established methodologies in previous work (Liu et al., 2023; Cabessa et al., 2025), we first structured the raw documents into the *Alpaca* format, which is particularly suitable for specializing LLMs via the *instruction-following* paradigm. Each ground-truth tagged document D_i^τ was structured as a JSON file containing three fields: an *instruction* I, an *input context* C, and the corresponding *expected output* Y, as illustrated in Appendix A.2.

Argument Component Identification Module (ACIM). For the ACS task, we obtain the ACIM as a fine-tuned LLM by optimizing the following loss function:

$$\mathcal{L}_{\text{ACS}}(\theta) = -\sum_{i=1}^{L}\sum_{t=1}^{T_i} \log P_\theta(Y_{i,t} \mid Y_{i,<t}, I_i, C_i) \quad (1)$$

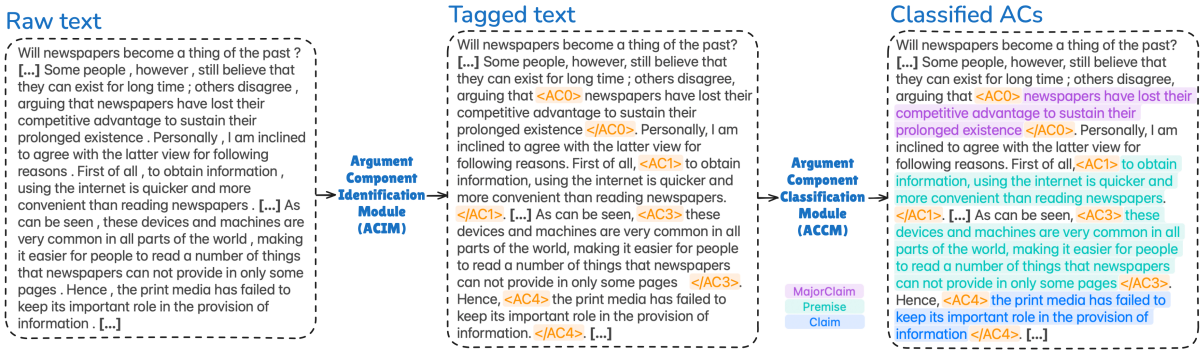

Figure 2: Workflow of our proposed approach: given a raw unstructured text, we perform fine-grained ACS by tagging via our Argument Identification Module; these can hence be utilized for the downstream ACC task.

where L is the total number of documents in the training set, T_i denotes the number of tokens in D_i^τ, and $Y_{i,t}$ is the t-th token in D_i^τ. Intuitively, this means that we want to optimize the set of parameters θ so that the considered LLM learns to properly insert the argument delimitation tags while preserving the original structure of the document.

Argument Component Classification Module (ACCM). For the ACC task, we obtain the ACCM, implementing the classification function γ, as a fine-tuned LLM by optimizing the following loss function:

$$\mathcal{L}_{\text{ACC}}(\theta) = -\sum_{i=1}^{L}\sum_{j=1}^{m_i} \log P_\theta\big(c_{i,j} \mid a_{i,j}\big) \qquad (2)$$

where L is the total number of document in the training set, m_i is the number of argument components in the i-th document, $a_{i,j}$ is the j-th component of document i, and $c_{i,j}$ is its corresponding gold label (e.g., *Premise*, *Claim*, or *Major Claim*). The goal is to train the model to classify each argument component based on its textual content, across all documents in the corpus. The full set of prompts utilized for fine-tuning our LLMs is reported in Appendix A.2.

4.2 Models

We utilize *small* and *open-weight* Large Language Models that are publicly available from the Huggingface Hub.[1] This choice is motivated by the possibility to train and use them under low-resource constraints with proper inference times, thus ensuring ease of access and scalability in AM tasks (Favero et al., 2025). These include *Llama3.1 8B Instruct*, *Qwen2.5 7B Instruct* and *Mistral v0.3 7B Instruct*. Details on how we deploy and fine-tune these models, as well as the main hyperparameters we used, are reported in Appendices A.2-A.4.

4.3 Datasets

We conduct our experimental analysis by resorting to two benchmark datasets for argument mining:

Persuasive Essays (PE) (Stab and Gurevych, 2017) containing 402 essays containing argument components that are annotated with their start/end positions, component types or classes (i.e., *Major Claim*, *Claim*, *Premise*).

AbstRCT (Mayer et al., 2020) containing 659 abstracts of Randomized Controlled Trials extracted from PubMed, spanning three main categories, i.e., neoplasms (neo), glaucomas (gla), and mixed (mix). Note that, for this dataset, we follow previous studies (Mayer et al., 2020; Liu et al., 2023; Cabessa et al., 2025) that consider *Claim* and *Major Claim* as a single *Claim* class, and *Evidence* as the *Premise* class. Additional details on datasets are reported in Appendix A.1.

In contrast with earlier approaches, we do not perform any preprocessing steps that would alter the original text of these datasets. In this regard, we avoided the manual insertion of additional structural tags beyond those required for delimiting argument components to facilitate segmentation (Cabessa et al., 2025), as we want our approach to operate under realistic, unstructured input conditions and avoid reliance on handcrafted cues that may not generalize across domains. Similarly, we avoided correcting grammatical errors in the original data (Favero et al., 2025), as we require our approach to faithfully adhere to the original text—regardless of the presence of typos.

[1] http://huggingface.co/models

4.4 Fine-tuning Strategies

We adopted the same prompting strategies proposed in Cabessa et al. (2025) for ACC, and employed a slightly adapted version of the prompt introduced in Favero et al. (2025) for ACS (cf. Appendix A.2). We resorted to the *QLoRA* strategy to maintain the fine-tuning efficient, using a 4-bit quantization of the model's pre-trained weights, and training a low-rank adaptation of these, thus significantly reducing memory consumption and computational requirements without sacrificing performance (Dettmers et al., 2023).

4.5 Post-processing Strategies

As both our ACS and ACC problems are conceived as text-generation tasks, it can occur that the considered LLMs might hallucinate (Huang et al., 2025), thus generating a different number of argument components compared to the ground truth.

To mitigate this issue, simply resorting to a matching between enumerated tag delimiters (i.e., matching $\langle AC_i \rangle$ from the ground truth with the one predicted, provided that i coincides) is not effective. Indeed, if the considered LLM generates a different number of components (being them less or more), we will lose our ability to perform exact matching.

Therefore, we perform so-called Best-Matching (BM) among candidate argument components by means of their word overlap, aiming at matching each ground truth component to the closest predicted one. A detailed overview of this process is reported in Algorithm 1. Note that, as hallucinations occur only rarely and to a negligible extent, we omit cases with zero overlap in Algorithm 1. Furthermore, we adopt a straightforward word-overlap approach to enhance efficiency, as our models are fine-tuned to preserve the original word order of components, thereby reducing the likelihood of misleading matches due to reordering.

Algorithm 1: Best Matching (BM)

Input : Predicted tagged text D^*; Ground-truth tagged text D^τ;
Output : Matched pairs $M \subseteq AC^* \times AC^\tau$

Step 1: ACs Extraction
Use regex to extract: $AC^* = \{x_1, ..., x_{m^*}\}$; $AC^\tau = \{y_1, ..., y_{m^\tau}\}$
Step 2: Pairing ACs
Initialize $M \leftarrow \emptyset$;
foreach $i \in [1, ..., m^*]$ **do**
  $k = 0; c_i = 0;$
  for $j \in [1, ..., m^\tau]$ **do**
    $overlap = \frac{|x_i \cap y_j|}{min\,(|x_i|, |y_j|)}$
    if $overlap > c_i$ **then**
      $c_i = overlap; k = j$;
    end
    if $overlap = 1$ *(i.e., exact matching)* **then**
      break
    end
  end
  $M \leftarrow M \cup \{(x_i, y_k)\}$;
  $AC^\tau \leftarrow AC^\tau \setminus \{y_j\}$;
end
return M

5 Evaluation Metrics

Next, we introduce the set of metrics we considered to evaluate the performances of our tested LLMs on the ACS and ACC tasks.

5.1 ACS Metrics

ACS requires measuring how accurately the predicted argument components, i.e., annotated by the fine-tuned LLMs, match with the human-provided ground truth annotations. To capture this, we resort to both *token-level labeling* as well as *argument-level* overlap and similarity metrics.

Given a collection of L documents $\mathcal{D} = \{D_1, D_2, ..., D_L\}$, we apply the LLM-based segmentation function f to obtain the predicted annotations $\mathcal{D}^* = \{D_1^*, D_2^*, ..., D_L^*\}$, where $D_i^* = f(D_i)$. From each annotated document D_i^*, we extract the set of predicted argument components, denoted by AC_i^*. Let us denote with D_i^τ the ground truth human annotations available for each document D_i, and with AC_i^τ the corresponding components. To evaluate annotation quality, we rely on the following set of established reliability metrics.

Correlation Over Argument Component Counts. As in Kasner et al. (2025), we first count how many argument components were annotated for each document in D_i^* compared to D_i^τ as:

$$\rho(\mathcal{D}^*, \mathcal{D}^\tau) = \rho\left(\langle |AC_i^*|, |AC_i^\tau| \rangle_{i=1,...,L}\right) \quad (3)$$

where ρ is the Pearson correlation coefficient, i.e., the higher the better.

BIO. (Ramshaw and Marcus, 1995) We convert both predicted and ground-truth documents into BIO-labeled sequences, where token beginning an argument, including the opening tag, are labeled as B (Begin), inner tokens within the boundaries of the argument are labeled as I (Inside), and tokens outside any argument component, including

the closing tag, are labeled as O (Outside). We compute the corresponding F_1 score based on the obtained labeled sequences. This score is particularly sensitive to boundary errors or segmentation mismatches, which can typically occur in LLM-based approaches (cf. Section 4.5).

Precision, Recall, and $\mathbf{F_1}$. We consider argument-level precision (P), recall (R), and F1-score metrics, as typically used in similar tasks (Da San Martino et al., 2019; Kasner et al., 2025), adapting them to the ACS task to measure the alignment between predicted and gold-standard argument component annotations as outlined next. Let us denote with M_i the sequence of matched pairs obtained by applying the Best Matching algorithm over a pair of tagged and ground-truth documents (D_i^*, D_i^τ). We can define document-level P and R as:

$$P(D_i^*, D_i^\tau) = \frac{1}{|M_i|} \sum_{(x_i,y_i)\in M_i} \frac{|x_i \cap y_i|}{|x_i|} \quad (4)$$

$$R(D_i^*, D_i^\tau) = \frac{1}{|M_i|} \sum_{(x_i,y_i)\in M_i} \frac{|x_i \cap y_i|}{|y_i|} \quad (5)$$

with $\cap$ indicating the word-level overlap between predicted and human-annotated components.

These scores are then aggregated for a tagged corpus $\mathcal{D}^*$ and its ground truth corpus $\mathcal{D}^\tau$ using a weighted average approach as follows:

$$P(\mathcal{D}^*, \mathcal{D}^\tau) = \frac{1}{\sum_i |M_i|} \sum_i P(D_i^*, D_i^\tau) \cdot |M_i| \quad (6)$$

$$R(\mathcal{D}^*, \mathcal{D}^\tau) = \frac{1}{\sum_i |M_i|} \sum_i R(D_i^*, D_i^\tau) \cdot |M_i| \quad (7)$$

Finally, we obtain the F_1-score as the harmonic mean between P and R.

ROUGE. Since the aforementioned metrics do not account for the sequential order of words within argument components, we also resort to ROUGE-L (Lin, 2004), which evaluates the longest common subsequence (LCS) between the predicted and reference spans, thereby capturing both content overlap and word order alignment.

POS Distribution Divergence. Beyond surface-level token overlap, we also deepened at the composition of predicted components, compared to the ground-truth ones, by analyzing their Part-of-Speech (POS) distributions. The rationale here is twofold: (i) POS allows us to gain insights into the syntactic role of included/excluded tokens, and (ii) we can investigate whether and to what extent the LLM is omitting uninformative or structurally irrelevant parts (e.g., determiners, conjunctions), or mistakenly excluding essential argumentative elements (e.g., nouns, verbs). To quantify this, we compute the *Kullback-Leibler divergence* as:

$$D_{KL}(P \,||\, Q) = \sum_i P(i) \log_2 \left(\frac{P(i)}{Q(i)}\right) \quad (8)$$

where P and Q represent the relative frequencies of POS tags in the i-th ground-truth and generated components, respectively. The lower the divergence, the higher the fidelity in capturing argument-relevant language.

5.2 ACC Metrics

For this task, we adopted standard evaluation metrics derived from the confusion matrix of the corresponding multi-class classification problem—namely, the macro-averaged *precision* (P), *recall* (R), and F_1*-score* (F_1).

6 Experimental Results

In this Section, we report and discuss the results we obtained for the Argument Component Segmentation task, unveiling whether and to what extent fine-tuned LLMs are able to match human expert annotations for argument components in raw texts. Furthermore, we investigate the impact of fully automated ACS in the downstream ACC task. Interested readers can refer to Figure 8 (Appendix B.3) for a qualitative analysis of our proposed approach.

6.1 Argument Component Segmentation

Tables 1-2 report the ACS performances of our considered LLMs across the datasets presented in Section 4.3. We next discuss them by metric.

Correlation Analysis. The correlation coefficient ρ in Table 1 shows a notable alignment between the number of predicted argument components and the ground-truth annotations. This is particularly evident for the PE and AbstRCT (gla) datasets, where models exhibit a very high correlation, consistently around 0.9, with Mistral being the more robust. Conversely, AbstRCT-*neo* and *mix*

Model (4bit)	Persuasive Essays ρ	Persuasive Essays *pred*	Persuasive Essays *gold*	AbstRCT (gla) ρ	AbstRCT (gla) *pred*	AbstRCT (gla) *gold*
Llama 3.1 8B	0.918	14.438		**0.878**	6.320	
Mistral v0.3 7B	**0.942**	14.528	*14.512*	0.872	6.320	*5.940*
Qwen 2.5 7B	0.908	14.738		0.868	6.430	

Model (4bit)	AbstRCT (neo) ρ	AbstRCT (neo) *pred*	AbstRCT (neo) *gold*	AbstRCT (mix) ρ	AbstRCT (mix) *pred*	AbstRCT (mix) *gold*
Llama 3.1 8B	0.679	5.970		**0.790**	6.630	
Mistral v0.3 7B	**0.712**	6.630	*6.860*	0.787	5.980	*6.000*
Qwen 2.5 7B	0.685	7.230		0.721	6.390	

Table 1: Pearson correlation and average number of predicted (pred) vs. ground-truth (gold) argument components across documents in the test datasets.

appear to be more challenging, achieving the highest correlations of 0.71 (Mistral) and 0.79 (Llama), respectively. These trends are further supported by the average number of predicted components, which closely matches the gold annotations (cf. Table 1). Most models tend to predict slightly more components—typically just one extra, on average—which can often be traced back to cases where a single gold component is split into two due to the filtering of conjunctions or discourse markers, leading the model to segment the argument more finely, as illustrated in Figure 8 (Appendix B.3).

BIO Tags. The BIO F_1 scores reported in Table 2 provide a stricter assessment of segmentation quality, as they reward only exact matches between predicted and gold-standard argument spans, reflecting the challenge of precise boundary alignment. The strongest performance is observed with Mistral on PE (0.910), Llama on AbstRCT-*gla* (0.914), and Mistral again on AbstRCT-*neo* and -*mix* (0.912 and 0.906, respectively). The corresponding IO values suggest that most discrepancies were due to differing B tag positions. Qualitative analysis indicated that LLMs tend to omit initial tokens they deem uninformative compared to human annotators, while still correctly identifying the core argumentative tokens, as shown in Figure 7.

KL-Divergence. This robustness is further supported by the Kullback-Leibler divergencies reported in Table 11 (Appendix B.2), which reveal no shift in Part-of-Speech distributions between predicted and ground-truth components. Notably, slightly higher discrepancies are observed in external tokens, i.e., those omitted by the LLM but included by human annotators, thus highlighting the tendency to filter out less informative elements.

Model (4bit)	Persuasive Essays *BIO*	*IO*	*P*	*R*	F_1	*R-L*
Llama 3.1 8B	0.710	0.857	0.907	0.907	0.906	**0.966**
Mistral v0.3 7B	**0.910**	**0.925**	**0.910**	**0.920**	**0.914**	**0.966**
Qwen 2.5 7B	0.888	0.908	0.908	0.918	0.912	0.960

Model (4bit)	AbstRCT (gla) *BIO*	*IO*	*P*	*R*	F_1	*R-L*
Llama 3.1 8B	**0.914**	**0.945**	**0.878**	**0.886**	**0.882**	**0.988**
Mistral v0.3 7B	0.909	0.938	0.873	0.884	0.878	0.986
Qwen 2.5 7B	0.912	0.942	0.877	0.875	0.875	0.986

Model (4bit)	AbstRCT (neo) *BIO*	*IO*	*P*	*R*	F_1	*R-L*
Llama 3.1 8B	0.898	0.927	0.875	**0.897**	0.885	0.981
Mistral v0.3 7B	**0.912**	**0.934**	0.876	**0.897**	**0.886**	**0.983**
Qwen 2.5 7B	0.893	0.917	**0.879**	0.889	0.883	0.980

Model (4bit)	AbstRCT (mix) *BIO*	*IO*	*P*	*R*	F_1	*R-L*
Llama 3.1 8B	0.904	0.940	0.878	**0.891**	**0.884**	0.984
Mistral v0.3 7B	**0.906**	**0.941**	0.876	0.887	0.880	**0.983**
Qwen 2.5 7B	0.899	0.934	**0.881**	0.885	0.881	0.981

Table 2: ACS results on the test datasets averaged over 20 runs. Best scores are bolded. *R-L* means ROUGE-L.

Precision, Recall, and F_1. Under overlap-based metrics, which allow for a more flexible matching, all models exhibit strong P and R, with an F_1 score consistently ranging between 0.87 and 0.92. This suggests that, despite occasional boundary mismatches, the core content of argument components is correctly preserved and predicted.

ROUGE-L. The quality of these core overlaps is further reinforced by the very high ROUGE-L scores, which reach the maximum for PE and are no lower than 0.99 across all AbstRCT subsets. These results underscore that, even when exact boundary matches are not achieved (as hinted by BIO scores), the predicted components maintain a highly faithful reproduction of the gold-standard content. Indeed, the preservation of the longest common subsequence (LCS) indicates that the models consistently capture essential lexical and structural parts of the arguments, ensuring high fidelity despite surface-level discrepancies.

Comparison with Baseline Approaches. To further validate the effectiveness of our proposed approach and models, we compare them against prompt-based baselines under zero-shot and few-shot settings. As shown in Table 3, there is a substantial performance gap, as the performance gains achieved by our method cannot be repro-

		PE		AbstRCT (avg)	
Model	**Type**	BIO	IO	BIO	IO
Llama 3.1 8B	FT	**0.710**	**0.857**	**0.905**	**0.937**
	FS	0.409	0.546	0.469	0.427
	ZS	0.284	0.414	0.388	0.317
Mistral v0.3 7B	FT	**0.910**	**0.925**	**0.909**	**0.938**
	FS	0.342	0.450	0.399	0.343
	ZS	0.170	0.254	0.228	0.327
Qwen 2.5 7B	FT	**0.888**	**0.908**	**0.901**	**0.931**
	FS	0.417	0.494	0.483	0.543
	ZS	0.182	0.268	0.224	0.335

Table 3: Comparison between Fine-tuning (FT), Few-shot (FS), and Zero-shot Prompting (ZS) in the ACS task. AbstRCT scores are aggregated over splits.

Dataset	Method	BIO
Persuasive Essays	***Ours (Mistral v0.3 7B)***	**0.910**
	Petasis (2019)	0.901
	Stab and Gurevych (2017)	0.867
	Morio et al. (2022)	0.852
AbstRCT (gla)	**Ours (Llama 3.1 8B)**	**0.914**
	Mayer et al. (2020)	0.870
	Morio et al. (2022)	0.703
AbstRCT (neo)	**Ours (Mistral v0.3 7B)**	**0.912**
	Mayer et al. (2020)	0.910
AbstRCT (mix)	Ours (Mistral v0.3 7B)	0.906
	Mayer et al. (2020)	**0.910**

Table 4: Comparison with competing methods on ACS.

duced through direct prompting alone, even under a few-shot setting, further strengthening the advantages of our proposed approach.

Comparison with Earlier Approaches. Table 4 shows that our proposed approach consistently outperforms prior works on ACS, or matches them in the most challenging cases. The observed gains, especially on complex datasets like AbstRCT (gla), support the suitability of fine-tuned LLMs for ACS, thanks to their greater contextual awareness compared to earlier approaches. Notably, these approaches typically leverage structural and syntactic features (Stab and Gurevych, 2017), contrasting with our more generalizable, generative setting.

Cross-domain Evaluation. We also performed a cross-domain evaluation, aiming to validate whether and to what extent models trained on a domain (e.g., essays) can generalize to other, unseen domains (e.g., scientific abstracts). As reported in Table 5, there is an asymmetry in generalization across domains. Models trained on essays maintain strong in-domain performance and still handle scientific abstracts reasonably well. In contrast,

		Test	
Model	**Train**	**PE**	**AbstRCT (avg)**
Llama 3.1 8B	PE	0.710	0.790
	AbstRCT	0.247	0.905
Mistral v0.3 7B	PE	0.910	0.786
	AbstRCT	0.214	0.909
Qwen 2.5 7B	PE	0.888	0.679
	AbstRCT	0.286	0.901

Table 5: Cross-domain BIO scores. Models are trained on one domain and tested across multiple, unseen domains. Training on AbstRCT is performed using the *neo* split only. AbstRCT values are averaged across splits.

models trained exclusively on scientific abstracts achieve excellent results within that domain but transfer very poorly to essays. We ascribe this contrast to two factors: (i) the prevalence of specialized technical vocabulary in AbstRCT compared to PE, and (ii) the fact that PE contains a richer and more explicit argumentative structure than AbstRCT. Accordingly, models trained on AbstRCT struggle to recognize the variety of argumentative structures in PE, resulting in a decrease in performance. Conversely, while models trained on PE might face difficulty in handling the technical language of AbstRCT, this impact is limited due to greater generalizability.

6.2 Argument Component Classification

To assess the impact of automated segmentation on the downstream ACC task, we fine-tuned our LLMs for ACC using the training split of each gold-standard dataset. For evaluation, we tested each model on two settings: one using human-annotated components and one using automatically segmented components. For the latter, we selected, for each dataset in Table 2, the output from the best-performing ACS model. This setup allows us to directly quantify the performance gap introduced by automated segmentation. The comparative results reported in Table 6 show that the performance of the downstream ACC task remains particularly stable when switching from human-annotated to automatically segmented argument components. Indeed, the drop in F_1 is negligible across all considered datasets and models, being consistently below 0.06, resp. 0.008, for PE, resp. AbstRCT. Interestingly, in some cases (e.g., Mistral on AbstRCT-*mix*), the automatically generated segments might also induce better ACC performances than the human-annotated ones ($\Delta < 0$).

Persuasive Essays							
	Human			Ours			
Model (4bit)	P	R	F_1	P	R	F_1	ΔF_1
Llama 3.1 8B	0.884	0.875	0.879	0.837	0.833	0.835	**0.044**
Mistral v0.3 7B	0.870	0.864	0.867	0.799	0.803	0.801	0.066
Qwen 2.5 7B	0.860	0.850	0.855	0.815	0.804	0.809	0.046
Morio et al.	–	–	0.796	–	–	0.666	0.130

AbstRCT (gla)							
	Human			Ours			
Model (4bit)	P	R	F_1	P	R	F_1	ΔF_1
Llama 3.1 8B	0.930	0.922	0.926	0.925	0.917	0.921	0.005
Mistral v0.3 7B	0.934	0.934	0.934	0.935	0.930	0.933	**0.001**
Qwen 2.5 7B	0.940	0.931	0.936	0.939	0.920	0.928	0.008
Morio et al.	–	–	0.676	–	–	0.450	0.226

AbstRCT (neo)							
	Human			Ours			
Model (4bit)	P	R	F_1	P	R	F_1	ΔF_1
Llama 3.1 8B	0.931	0.920	0.925	0.930	0.916	0.923	0.002
Mistral v0.3 7B	0.938	0.932	0.935	0.934	0.928	0.931	0.004
Qwen 2.5 7B	0.940	0.933	0.936	0.954	0.935	0.944	**-0.008**

AbstRCT (mix)							
	Human			Ours			
Model (4bit)	P	R	F_1	P	R	F_1	ΔF_1
Llama 3.1 8B	0.949	0.941	0.945	0.949	0.942	0.945	0.000
Mistral v0.3 7B	0.946	0.949	0.947	0.955	0.958	0.957	**-0.010**
Qwen 2.5 7B	0.952	0.949	0.951	0.953	0.946	0.950	0.001

Table 6: ACC results on the test datasets averaged over 5 runs. ΔF_1 is the gain in F_1 by humans over LLMs. For Morio et al. we report the values as in their paper.

Notably, these results contrast with previous work where automated ACS is found to cause a strong reduction in ACC performances (Ding et al., 2023; Favero et al., 2025). For example, Morio et al. (2022) observed a reduction in F_1 of 0.130 and 0.226, considering the PE and AbstRCT-*gla* datasets, respectively. Conversely, we demonstrate that our automated paired-tag annotation schema performed via fine-tuned LLMs produces segmentations of high quality, leading to reliable ACC.

7 Conclusions

Argument Mining has advanced significantly in recent years, but a major bottleneck remains: most pipelines still depend on limited, manually annotated datasets, an approach that is costly, unscalable, and restricts broader adoption. At the same time, robust methods for automating Argument Component Segmentation are still lacking.

In this work, we address this gap by introducing a fine-grained annotation schema for reliable segmentation, implemented through fine-tuned LLMs that match human-level segmentation quality while offering scalability and cost-efficiency.

We evaluated its impact on the downstream Argument Component Classification, finding no drop in performance compared to human-labeled data, thus demonstrating that automated segmentation is feasible and properly supports downstream tasks, paving the way for truly end-to-end pipelines.

Future work will extend our framework to tasks like relation extraction and classification, aiming to fully automate the Argument Mining pipeline.

Acknowledgements

This work was supported by projects FAIR (PE0000013) and SERICS (PE00000014), under the MUR National Recovery and Resilience Plan funded by the EU - NextGenerationEU; and by the Italian Ministry of University and Research (MUR) PRIN 2022 grant 2022XERWK9 "SPICACHU - Semantics-based Provenance, Integrity, and Curation for Consistent, High-quality, and Unbiased data science" - CUP: H53C24000990006.

Limitations

Resource Language. Our current work focuses exclusively on English texts. However, linguistic variation across languages can significantly affect segmentation performance. As a next step, we plan to explore multilingual corpora and language models to extend our approach to multilingual setting.

Comparison with Prior Work. A direct comparison with existing works is currently limited by the lack of publicly available code or pre-trained models from prior work. To partially mitigate this, we deliberately resorted to the same datasets, and adopted the same prompting strategies described in those studies to ensure a fair yet indirect comparison under similar conditions.

Ethics Statement

Broader Impact. The main goal of our research is to advance the field of Argument Mining by providing a robust and scalable framework for automating Argument Component Segmentation (ACS). We acknowledge that our work may be used in downstream tasks that might have a societal impact (e.g., decision-making systems), and therefore we discard any responsibility for misuses and stress the importance of responsible and ethical use of these technologies by all actors involved.

Reproducibility. *We are committed to releasing all resources for this work upon request.*

References

Jérémie Cabessa, Hugo Hernault, and Umer Mushtaq. 2024. In-context learning and fine-tuning gpt for argument mining. *arXiv preprint arXiv:2406.06699*.

Jérémie Cabessa, Hugo Hernault, and Umer Mushtaq. 2025. Argument mining with fine-tuned large language models. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 6624–6635, Abu Dhabi, UAE. Association for Computational Linguistics.

Elena Cabrio and Serena Villata. 2018. Five years of argument mining: a data-driven analysis. In *International Joint Conference on Artificial Intelligence*.

Lucas Carstens and Francesca Toni. 2015. Towards relation based argumentation mining. In *Proceedings of the 2nd Workshop on Argumentation Mining*, pages 29–34, Denver, CO. Association for Computational Linguistics.

Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Linyi Yang, Kaijie Zhu, Hao Chen, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, Wei Ye, Yue Zhang, Yi Chang, Philip S. Yu, Qiang Yang, and Xing Xie. 2024. A survey on evaluation of large language models. *ACM Trans. Intell. Syst. Technol.*, 15(3).

Guizhen Chen, Liying Cheng, Anh Tuan Luu, and Lidong Bing. 2024. Exploring the potential of large language models in computational argumentation. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2309–2330, Bangkok, Thailand. Association for Computational Linguistics.

Zaiqian Chen, Daniel Verdi do Amarante, Jenna Donaldson, Yohan Jo, and Joonsuk Park. 2022. Argument mining for review helpfulness prediction. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 8914–8922, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Giovanni Da San Martino, Seunghak Yu, Alberto Barrón-Cedeño, Rostislav Petrov, and Preslav Nakov. 2019. Fine-grained analysis of propaganda in news articles. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5636–5646, Hong Kong, China. Association for Computational Linguistics.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. Qlora: Efficient finetuning of quantized llms. *Preprint*, arXiv:2305.14314.

Yuning Ding, Marie Bexte, and Andrea Horbach. 2022. Don‘t drop the topic - the role of the prompt in argument identification in student writing. In *Proceedings of the 17th Workshop on Innovative Use of NLP for Building Educational Applications (BEA 2022)*, pages 124–133, Seattle, Washington. Association for Computational Linguistics.

Yuning Ding, Marie Bexte, and Andrea Horbach. 2023. Score it all together: A multi-task learning study on automatic scoring of argumentative essays. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 13052–13063, Toronto, Canada. Association for Computational Linguistics.

Steffen Eger, Johannes Daxenberger, and Iryna Gurevych. 2017. Neural end-to-end learning for computational argumentation mining. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11–22, Vancouver, Canada. Association for Computational Linguistics.

Lucile Favero, Juan Antonio Pérez-Ortiz, Tanja Käser, and Nuria Oliver. 2025. Leveraging small llms for argument mining in education: Argument component identification, classification, and assessment. *Preprint*, arXiv:2502.14389.

Deniz Gorur, Antonio Rago, and Francesca Toni. 2025. Can large language models perform relation-based argument mining? In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 8518–8534, Abu Dhabi, UAE. Association for Computational Linguistics.

Ivan Habernal and Iryna Gurevych. 2017. Argumentation mining in user-generated web discourse. *Computational Linguistics*, 43(1):125–179.

Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, and Ting Liu. 2025. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *ACM Trans. Inf. Syst.*, 43(2).

Omid Kashefi, Sophia Chan, and Swapna Somasundaran. 2023. Argument detection in student essays under resource constraints. In *Proceedings of the 10th Workshop on Argument Mining*, pages 64–75, Singapore. Association for Computational Linguistics.

Zdeněk Kasner, Vilém Zouhar, Patrícia Schmidtová, Ivan Kartáč, Kristỳna Onderková, Ondřej Plátek, Dimitra Gkatzia, Saad Mahamood, Ondřej Dušek, and Simone Balloccu. 2025. Large language models as span annotators. *arXiv preprint arXiv:2504.08697*.

Masayuki Kawarada, Tsutomu Hirao, Wataru Uchida, and Masaaki Nagata. 2024. Argument mining as a text-to-text generation task. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2002–2014, St. Julian's, Malta. Association for Computational Linguistics.

Christian Kirschner, Judith Eckle-Kohler, and Iryna Gurevych. 2015. Linking the thoughts: Analysis of argumentation structures in scientific publications. In *Proceedings of the 2nd Workshop on Argumentation Mining*, pages 1–11, Denver, CO. Association for Computational Linguistics.

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

Ran Levy, Yonatan Bilu, Daniel Hershcovich, Ehud Aharoni, and Noam Slonim. 2014. Context dependent claim detection. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 1489–1500, Dublin, Ireland. Dublin City University and Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Boyang Liu, Viktor Schlegel, Riza Batista-Navarro, and Sophia Ananiadou. 2023. Argument mining as a multi-hop generative machine reading comprehension task. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 10846–10858, Singapore. Association for Computational Linguistics.

Tobias Mayer, Elena Cabrio, and Serena Villata. 2020. Transformer-based argument mining for healthcare applications. In *ECAI 2020 - 24th European Conference on Artificial Intelligence*, volume 325 of *Frontiers in Artificial Intelligence and Applications*, pages 2108–2115. IOS Press.

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

Vlad Niculae, Joonsuk Park, and Claire Cardie. 2017. Argument mining with structured SVMs and RNNs. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 985–995, Vancouver, Canada. Association for Computational Linguistics.

Raquel Mochales Palau and Marie-Francine Moens. 2009. Argumentation mining: the detection, classification and structure of arguments in text. In *Proceedings of the 12th International Conference on Artificial Intelligence and Law*, ICAIL '09, page 98–107, New York, NY, USA. Association for Computing Machinery.

Raquel Mochales Palau and Marie-Francine Moens. 2011. Argumentation mining. *Artif. Intell. Law*, 19(1):1–22.

Andreas Peldszus. 2014. Towards segment-based recognition of argumentation structure in short texts. In *Proceedings of the First Workshop on Argumentation Mining*, pages 88–97, Baltimore, Maryland. Association for Computational Linguistics.

Georgios Petasis. 2019. Segmentation of argumentative texts with contextualised word representations. In *Proceedings of the 6th Workshop on Argument Mining*, pages 1–10, Florence, Italy. Association for Computational Linguistics.

Mircea-Luchian Pojoni, Lorik Dumani, and Ralf Schenkel. 2023. Argument-mining from podcasts using chatgpt. In *ICCBR Workshops*, pages 129–144.

Peter Potash, Alexey Romanov, and Anna Rumshisky. 2017. Here's my point: Joint pointer architecture for argument mining. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1364–1373, Copenhagen, Denmark. Association for Computational Linguistics.

Lance Ramshaw and Mitch Marcus. 1995. Text chunking using transformation-based learning. In *Third Workshop on Very Large Corpora*.

Chris Reed, Raquel Mochales Palau, Glenn Rowe, and Marie-Francine Moens. 2008. Language resources for studying argument. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco. European Language Resources Association (ELRA).

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)*, pages 46–56, Doha, Qatar. Association for Computational Linguistics.

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

Purin Sukpanichnant, Anna Rapberger, and Francesca Toni. 2024. Peerarg: Argumentative peer review with llms. *arXiv preprint arXiv:2409.16813*.

Fan Zhang and Diane Litman. 2016. Using context to predict the purpose of argumentative writing revisions. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1424–1430, San Diego, California. Association for Computational Linguistics.

Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, and Zheyan Luo. 2024. LlamaFactory: Unified efficient fine-tuning of 100+ language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, pages 400–410, Bangkok, Thailand. Association for Computational Linguistics.

A Additional Details on Methodology

A.1 Data

Tables 7-8 provide details on the train/test splits as well as the number of components for each dataset used in our work.

Split	Essays
Train	322
Test	80
Total	402

Component Class	Count
Major Claim	751
Claim	1,506
Premise	3,832
Total	6,089

Table 7: Statistics (left) and component details (right) for the *Persuasive Essays* dataset.

	Split	Abstracts	Components
Train	*neo*	350	2291
Test	*neo*	100	691
	gla	100	615
	mix	100	609

Table 8: Statistics for the *AbstRCT* dataset.

A.2 Prompts

Figures 3–4 show the prompt templates defined for the Argument Component Segmentation task and the corresponding Alpaca format for the fine-tuning process on the Persuasive Essays (PE) dataset.

Likewise, Figures 5–6 report the templates we used for the Argument Component Classification task. Note that these templates were slightly adjusted to accommodate the specific formatting of each dataset outlined in Section 4.3.

A.3 Models

To ease reproducibility, Table 9 reports the HuggingFace tags for the models used in our study.

Name	HuggingFace ID
Llama 3.1 8B	`meta-llama-3.1-8b-instruct-bnb-4bit`
Mistral v0.3 7B	`mistral-7b-instruct-v0.3-bnb-4bit`
Qwen 2.5 7B	`Qwen2.5-7B-Instruct-unsloth-bnb-4bit`

Table 9: Detailed references for the (4 bit) models used in this study.

ACS Prompt

#TASK: Segment the following essay into distinct argument components. At the start of each argument component, insert the marker $\langle AC^* \rangle$. At the end of each argument component, insert the marker $\langle /AC^* \rangle$.

* is a sequential enumeration that starts from 0. Keep the original text in the same order without adding, removing, or altering any words (other than inserting the $\langle AC^* \rangle \langle /AC^* \rangle$ markers).

#GUIDELINES: Identify each coherent segment that forms a logical unit of the argument (e.g., claims, premises, evidence, or conclusions).

Figure 3: Example of prompt for the Argument Component Segmentation task on the PE dataset.

ACS Alpaca

{"**instruction**": "#TASK: Segment the following essay into distinct argument components. [...]",

"**input**": "From this point of view , I firmly believe that we should attach more importance to cooperation during primary education . [...]",

"**output**": "From this point of view , I firmly believe that <AC0> we should attach more importance to cooperation during primary education </AC0> . [...]"}

Figure 4: Example of Alpaca format for fine-tuning over the Argument Component Segmentation task on the PE dataset.

ACC Prompt

You are an expert in Argument Mining. You are given an essay which contains numbered argument components enclosed by $\langle AC^* \rangle \langle /AC^* \rangle$ tags.

Your task is to classify each argument components in the essay as either **Major Claim**, **Claim**, or **Premise**.

You must return a list of argument component types in the following JSON format:
{component_types: [component_type (str), component_type (str), ..., component_type (str)] }

Figure 5: Example of prompt for the Argument Component Classification task on the PE dataset.

ACC Alpaca

{"**instruction**": "### You are an expert in Argument Mining. You are given an essay [...]",

"**input**": "### Here is the essay text: Should students be taught to compete or to cooperate? [...]",

"**output**": "{component_types: [MajorClaim, Claim, Premise, ...]}"}

Figure 6: Example of Alpaca format for fine-tuning over the Argument Component Classification task on PE.

Parameter	Value
Train epochs	5
Train batch size	2
Gradient accumulation steps	4
Learning rate	5e-5
LR scheduler type	cosine
Warmup ratio	0.1
Max grad norm	1.0
Finetuning type	lora
LoRA target	all
Quantization bit	4
LoRA + LR ratio	16.0
FP16	True
Temperature	Model default

Table 10: Fine-tuning parameters for our LLMs.

A.4 Hyperparameters

All models were fine-tuned through the *LLaMA-Factory* Python library (Zheng et al., 2024),[2] using the hyperparameters reported in Table 10, on a single consumer NVIDIA RTX 3090 (24GB) GPU.

B Additional Details on Results

B.1 Tagging Mismatches

Figure 7 reports an example of a mismatch between ground-truth (top) and predicted (bottom) arguments due to the filtering out of tokens deemed as potentially uninformative.

B.2 Part-of-Speech Evaluation

Table 11 reports Part-of-Speech tagging KL divergencies, which serve as a proxy for consistency during the Argument Component Segmentation process as they reflect how well the model preserves the original syntactic structure during segmentation. Notably, across datasets, we observe extremely low divergencies, suggesting that fine-tuned models learn to insert paired tags to delimit

[2] https://github.com/hiyouga/LLaMA-Factory

<AC3> As can be seen , these devices and machines are very common in all parts of the world , making it easier for people to read a number of things that newspapers can not provide in only some pages </AC3> .

As can be seen, <AC3> these devices and machines are very common in all parts of the world, making it easier for people to read a number of things that newspapers can not provide in only some pages </AC3>.

Figure 7: Example of mismatch between ground-truth (top) and predicted (bottom) arguments due to potentially uninformative tokens.

Persuasive Essays		
Model (4bit)	*External*	*Internal*
Llama 3.1 8B	$4.81 \times e^{-3}$	$5.16 \times e^{-3}$
Mistral v0.3 7B	$5.92 \times e^{-4}$	$1.41 \times e^{-4}$
Qwen 2.5 7B	$1.19 \times e^{-3}$	$1.58 \times e^{-4}$

AbstRCT		
Model (4bit)	*External*	*Internal*
Llama 3.1 8B (gla)	$2.20 \times e^{-3}$	$3.50 \times e^{-4}$
Llama 3.1 8B (neo)	$5.23 \times e^{-4}$	$4.73 \times e^{-4}$
Llama 3.1 8B (mix)	$1.97 \times e^{-3}$	$2.74 \times e^{-4}$
Mistral v0.3 7B (gla)	$2.26 \times e^{-3}$	$4.17 \times e^{-4}$
Mistral v0.3 7B (neo)	$4.00 \times e^{-4}$	$3.32 \times e^{-4}$
Mistral v0.3 7B (mix)	$1.88 \times e^{-3}$	$1.75 \times e^{-4}$
Qwen 2.5 7B (gla)	$2.07 \times e^{-3}$	$2.73 \times e^{-4}$
Qwen 2.5 7B (neo)	$4.55 \times e^{-4}$	$3.49 \times e^{-4}$
Qwen 2.5 7B (mix)	$3.19 \times e^{-4}$	$2.26 \times e^{-4}$

Table 11: KL divergencies for POS-tagging on the test datasets. External, resp. internal, refer to the tokens outside, resp. inside, the argument components.

argument components without disrupting the linguistic integrity of the raw input text.

B.3 Qualitative ACS and ACC Example

Figure 8 provides a comparison of the ACS and subsequent ACC tasks between human annotators and LLMs on a sample from Persuasive Essays.

Human Annotation – Human Classification

The expression " Never , never give up " means keep trying and never stop working Many individuals throughout their life face with at least one failure in different areas such as education , love and economy . Some believe when a person faces with a failure the best way is to forget that and releases that ; however , I completely disagree with this idea . I always believe that <AC0> a defeated person has to try and try until to achieve her or his goal </AC0> . First of all , <AC1> when people face with a failure they can learn a large number of things from that failure , therefore in the next effort they try to avoid doing and repeating that mistakes </AC1> . For instance , <AC2> most of inventors in the first phases of their job face with some problems and defeats , but in the next stages they learn how to deal with problems </AC2> . Second of all , <AC3> experience shows while people doing a job constantly , they become more experienced in that subject , so in the final effort they become successful with the high percentile </AC3> . For example , <AC4> scientists experiment a substance for a long time , after they gain lots of experience and knowledge about that subject they decide to make it final </AC4> . To sum up , I would maintain that <AC5> people should not give up when they face with a failure </AC5> . In opposite , <AC6> they have to learn new points from that failure , and they become more experienced </AC6> .

{"component_types": ["MajorClaim", "Claim", "Premise", "Claim", "Premise", "MajorClaim", "Claim"]}

LLM Annotation – LLM Classification

The expression " Never, never give up " means keep trying and never stop working Many individuals throughout their life face with at least one failure in different areas such as education, love and economy. Some believe when a person faces with a failure the best way is to forget that and releases that ; however, I completely disagree with this idea. I always believe that <AC0> a defeated person has to try and try until to achieve her or his goal </AC0>. First of all, <AC1> when people face with a failure they can learn a large number of things from that failure </AC1>, therefore <AC2> in the next effort they try to avoid doing and repeating that mistakes </AC2>. For instance, <AC3> most of inventors in the first phases of their job face with some problems and defeats, but in the next stages they learn how to deal with problems </AC3>. Second of all, <AC4> experience shows while people doing a job constantly, they become more experienced in that subject </AC4>, so <AC5> in the final effort they become successful with the high percentile </AC5>. For example, <AC6> scientists experiment a substance for a long time, after they gain lots of experience and knowledge about that subject they decide to make it final </AC6>. To sum up, I would maintain that <AC7> people should not give up when they face with a failure </AC7>. In opposite, <AC8> they have to learn new points from that failure, and they become more experienced </AC8>.

{"component_types": ["MajorClaim", "Premise", "Claim", "Premise", "Premise", "Claim", "Premise", "MajorClaim", "Claim"]}

Figure 8: Human- (top) vs LLMs-based (bottom) ACS and subsequent ACC tasks on the PE dataset.