ARGUS: Towards End-to-End Argument Mining with Large Language Models

Ettore Caputo, Sergio Greco, Lucio La Cava

DIMES Department, University of Calabria, Italy
{ettore.caputo, s.greco, lucio.lacava}@dimes.unical.it

Abstract

We present ARGUS, an end-to-end Argument Mining (AM) tool that exploits Large Language Models (LLMs) to automatically perform all core AM tasks, i.e., Argument Component Segmentation, Classification, Relation Identification, and Relation Classification. Furthermore, ARGUS builds the corresponding argumentation framework (AF) and seamlessly integrates symbolic solvers to compute extensions and perform formal reasoning. ARGUS is designed to ensure broad flexibility and usability, supporting any open-source or commercial LLMs and symbolic solvers, providing a ready-to-use platform for exploring neuro-symbolic approaches to argumentation in both research and practical applications.

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). At its core, AM seeks to identify argumentative components (e.g., *premises* and *claims*) and the relations between them (e.g., *attacks* and *supports*). These components can be formalized into *abstract argumentation frameworks* (AFs) (Dung 1995) to provide a symbolic representation of debates and enable reasoning over conflicting perspectives. Such representations are increasingly valuable in fields that require reasoning under disagreement or uncertainty, including legal analysis (Palau and Moens 2009), scientific debate (Sukpanichnant, Rapberger, and Toni 2024), and web discourse (Habernal and Gurevych 2017).

Argument Mining pipelines typically consist of four key subtasks (Eger, Daxenberger, and Gurevych 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 argumentative component (e.g., claim, premise); (iii) *Argument Relation Identification* (ARI), to detect relations among argumentative components; and (iv) *Argument Relation Classification* (ARC) to label the detected relations (e.g., attacks, supports). Together, these steps allow the construction of an argumentation framework in the form of a graph that symbolic solvers can process to compute extensions and derive conclusions through formal reasoning. While this conceptual pipeline is well established, existing approaches typically address sub-tasks in isolation, using 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, Park, and Cardie 2017), optimization techniques (Stab and Gurevych 2017), or deep neural networks like RNNs (Eger, Daxenberger, and Gurevych 2017; Niculae, Park, and Cardie 2017) and LSTMs (Potash, Romanov, and Rumshisky 2017). This fragmentation stems from the difficulty of such architectures in modeling argumentative discourse, which requires capturing implicit reasoning, contextual dependencies, and subtle linguistic cues, thus preventing the development of integrated frameworks that can automatically execute the entire AM pipeline end-to-end, from raw text to symbolic reasoning over AFs.

Recently, LLMs have shown strong potential for advancing and automating the Argument Mining pipeline. Indeed, their remarkable Natural Language Understanding capabilities (Chang et al. 2024) allow for a flexible and data-efficient processing, requiring little to no task-specific data compared to previous architectures, leveraging in-context learning or fine-tuning. This versatility has been shown to be effective for various AM subtasks (Chen et al. 2024; Cabessa, Hernault, and Mushtaq 2025), including both the identification and classification of argument components (Favero et al. 2025; Cabessa, Hernault, and Mushtaq 2024) and the detection and labeling of relations among them (Pojoni, Dumani, and Schenkel 2023; Gorur, Rago, and Toni 2025).

Despite these advances, to date, no existing tool fully realizes this potential in an integrated framework that unifies LLM-based AM with the construction of argumentation frameworks and subsequent symbolic reasoning.

Contributions In this paper, we present ARGUS, the first tool aiming to fill this gap. ARGUS leverages LLMs to perform all AM tasks individually, whether users only need one sub-task, or sequentially—from raw text to the construction of the corresponding AF and subsequent symbolic solving. ARGUS is conceived to be used with any existing open-source or commercial LLM, whether general-purpose or fine-tuned for task-specific applications.

The ARGUS Tool

Let $D = [t_1, t_2, ..., t_n]$ represent an open-ended textual document as a sequence of tokens, where each t_i is a word drawn from a vocabulary $\mathcal{V}$. The goal of ARGUS is to perform all key steps of Argument Mining and eventually map D into a formal argumentation framework (AF) (Dung 1995). An integrated symbolic solver can then process the obtained AF to compute one or more *extensions* under a chosen *argumentation semantics*, enabling formal reasoning over the extracted argumentative structure.

Argument Mining Pipeline

We formalize the LLM-based Argument Mining pipeline in ARGUS as a function f_θ parametrized by a set of parameters θ, i.e., $f_\theta(D) = \langle A, R, S \rangle$, where $D = [t_1, ..., t_n]$ is a document and $\langle A, R, S \rangle$ is a (bipolar) argumentation framework, obtained through a sequence of intermediate steps, as illustrated next. In particular, $A = \{a_1, ..., a_k\}$ $(k \leq n)$ is a set of arguments derived from D in such a way that (i) for all $i \leq k$, a_i denotes a sub-sequence of terms in D (called argument), and (ii) for all $i < k$, terms associated with a_i precede terms associated with a_{i+1} in D. (i.e. sequences associated with two distinct arguments do not overlap), whereas R and S denote relations among arguments, that is *attacks (R)* and *supports (S)*, respectively. For the sake of readability, we denote this process as a single function f_θ, despite it may correspond either to a single pre-trained LLM applied to all steps or task-specific LLMs for each step—each with its own set of parameters θ. Note that ARGUS is designed to keep humans-in-the-loop, allowing them to skip entire stages (e.g., if arguments are already known or given) or manually refine the outputs of each step (e.g., adding an overlooked argument), ensuring that domain expertise remains in the pipeline.

Argument Component Segmentation The first stage of the ARGUS pipeline isolates *argumentative* from *non-argumentative* spans in the input document D, grouping them into coherent components. Formally, the underlying LLM maps D into a set of arguments $A = \{a_1, ..., a_k\}$, where each a_i corresponds to a contiguous subsequence of tokens in D, delimited by the tags $\langle AC_i \rangle$ and $\langle /AC_i \rangle$.

Argument Component Classification This step leverages the underlying LLM to assign each extracted argument in A with a label $l \in L$, indicating its argumentative role. By default, we adopt $L = \{Premise, Claim, Major\ Claim\}$, ensuring consistency with established conventions in existing AM resources (Cabessa, Hernault, and Mushtaq 2025; Favero et al. 2025). Nonetheless, ARGUS allows users to fully customize L, i.e., to define alternative or domain-specific argumentative roles.

Arguments Relation Identification and Classification Given the set of arguments A and the associated labellings (computed at the previous step), the underlying LLM outputs two relations $R \subseteq A \times A$ (denoting attacks) and $S \subseteq A \times A$ (denoting supports).

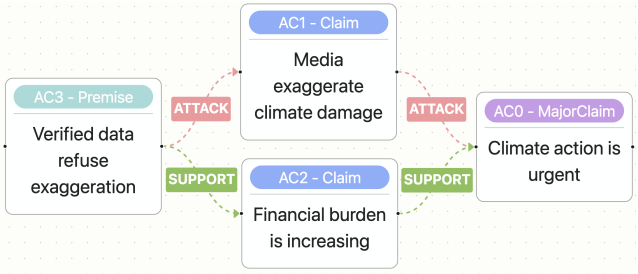

Figure 1: An Example of BAF generated by ARGUS.

Argumentation Framework Creation and Solving Finally, the ARGUS pipeline constructs the bipolar argumentation framework $\Lambda = \langle A, R, S \rangle$ (Dung 1995; Amgoud et al. 2008; Cohen et al. 2014), as shown in Figure 1. Once Λ is built, ARGUS seamlessly interacts with symbolic solvers to compute the corresponding set of extensions, obtained under a given semantics σ available in the chosen solver, e.g., *complete*, *grounded*, *preferred*, and *stable* (Dung 1995). Each extension E denotes a set of arguments which together can be assumed as true elements, under the given semantics σ.

Implementative Details

ARGUS uses the *transformers* Python library for model serving. By default, it loads task-specific fine-tuned LLMs, while its modular design ensures full compatibility with alternative models. This design ensures loading small and efficient models locally, ensuring cost-effectiveness, low-latency, and privacy preservation when processing sensitive data (e.g., medical records). ARGUS also works with commercial LLMs (e.g., GPT) through the setting of service-specific API keys in the configuration panel.

To enable symbolic reasoning, ARGUS integrates with *Aspartix* (Dvorák et al. 2020), which supports a wide range of semantics for both Dung-style and Bipolar AFs. Note that the solver module is fully pluggable, allowing users to replace it with alternative solvers if needed.

Expected Impact

We anticipate that ARGUS will have a tangible impact on both researchers and practitioners by addressing a key gap in computational argumentation: it provides the first end-to-end tool for performing Argument Mining, up to symbolic reasoning. Notably, ARGUS keeps humans in the loop, enabling complementarity between the contextual awareness of LLMs, the precision of symbolic reasoning, and domain-expert knowledge. We expect ARGUS to facilitate the development of novel resources by mitigating the high costs and time associated with manual annotation (Lawrence and Reed 2019), as well as the development of new models that enhance the integration between LLMs and argumentation.

Release and Demonstration ARGUS *will be released upon request to the authors, and a commented screen recording of* ARGUS *in action is provided with this paper.*[1]

[1]https://drive.google.com/drive/folders/1Mi1sFO380fCBpZw8-PvV5ml34AQoWlnC

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.

References

Amgoud, L.; Cayrol, C.; Lagasquie-Schiex, M.; and Livet, P. 2008. On bipolarity in argumentation frameworks. *Int. J. Intell. Syst.*, 23(10): 1062–1093.

Cabessa, J.; Hernault, H.; and Mushtaq, U. 2024. In-context learning and fine-tuning GPT for argument mining. *arXiv preprint arXiv:2406.06699*.

Cabessa, J.; Hernault, H.; and Mushtaq, U. 2025. Argument Mining with Fine-Tuned Large Language Models. In *Proceedings of the 31st International Conference on Computational Linguistics*, 6624–6635. Abu Dhabi, UAE: Association for Computational Linguistics.

Cabrio, E.; and Villata, S. 2018. Five Years of Argument Mining: a Data-driven Analysis. In *International Joint Conference on Artificial Intelligence*.

Chang, Y.; Wang, X.; Wang, J.; Wu, Y.; Yang, L.; Zhu, K.; Chen, H.; Yi, X.; Wang, C.; Wang, Y.; Ye, W.; Zhang, Y.; Chang, Y.; Yu, P. S.; Yang, Q.; and Xie, X. 2024. A Survey on Evaluation of Large Language Models. *ACM Trans. Intell. Syst. Technol.*, 15(3).

Chen, G.; Cheng, L.; Luu, A. T.; and Bing, L. 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)*, 2309–2330. Bangkok, Thailand: Association for Computational Linguistics.

Cohen, A.; Gottifredi, S.; García, A. J.; and Simari, G. R. 2014. A survey of different approaches to support in argumentation systems. *Knowl. Eng. Rev.*, 29(5): 513–550.

Dung, P. M. 1995. On the Acceptability of Arguments and its Fundamental Role in Nonmonotonic Reasoning, Logic Programming and n-Person Games. *Artif. Intell.*, 77(2): 321–358.

Dvorák, W.; Rapberger, A.; Wallner, J. P.; and Woltran, S. 2020. ASPARTIX-V19 - An Answer-Set Programming Based System for Abstract Argumentation. In *Foundations of Information and Knowledge Systems - 11th International Symposium, FoIKS 2020, Dortmund, Germany, February 17-21, 2020, Proceedings*, volume 12012 of *Lecture Notes in Computer Science*, 79–89. Springer.

Eger, S.; Daxenberger, J.; and Gurevych, I. 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)*, 11–22. Vancouver, Canada: Association for Computational Linguistics.

Favero, L.; Pérez-Ortiz, J. A.; Käser, T.; and Oliver, N. 2025. Leveraging Small LLMs for Argument Mining in Education: Argument Component Identification, Classification, and Assessment. arXiv:2502.14389.

Gorur, D.; Rago, A.; and Toni, F. 2025. Can Large Language Models perform Relation-based Argument Mining? In *Proceedings of the 31st International Conference on Computational Linguistics*, 8518–8534. Abu Dhabi, UAE: Association for Computational Linguistics.

Habernal, I.; and Gurevych, I. 2017. Argumentation Mining in User-Generated Web Discourse. *Computational Linguistics*, 43(1): 125–179.

Lawrence, J.; and Reed, C. 2019. Argument Mining: A Survey. *Computational Linguistics*, 45(4): 765–818.

Levy, R.; Bilu, Y.; Hershcovich, D.; Aharoni, E.; and Slonim, N. 2014. Context Dependent Claim Detection. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, 1489–1500. Dublin, Ireland: Dublin City University and Association for Computational Linguistics.

Niculae, V.; Park, J.; and Cardie, C. 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)*, 985–995. Vancouver, Canada: Association for Computational Linguistics.

Palau, R. M.; and Moens, M. 2011. Argumentation mining. *Artif. Intell. Law*, 19(1): 1–22.

Palau, R. M.; and Moens, M.-F. 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, 98–107. New York, NY, USA: Association for Computing Machinery. ISBN 9781605585970.

Pojoni, M.-L.; Dumani, L.; and Schenkel, R. 2023. Argument-Mining from Podcasts Using ChatGPT. In *IC-CBR Workshops*, 129–144.

Potash, P.; Romanov, A.; and Rumshisky, A. 2017. Here's My Point: Joint Pointer Architecture for Argument Mining. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 1364–1373. Copenhagen, Denmark: Association for Computational Linguistics.

Stab, C.; and Gurevych, I. 2014. Identifying Argumentative Discourse Structures in Persuasive Essays. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 46–56. Doha, Qatar: Association for Computational Linguistics.

Stab, C.; and Gurevych, I. 2017. Parsing Argumentation Structures in Persuasive Essays. *Computational Linguistics*, 43(3): 619–659.

Sukpanichnant, P.; Rapberger, A.; and Toni, F. 2024. PeerArg: Argumentative Peer Review with LLMs. *arXiv preprint arXiv:2409.16813*.