Proceedings of the Second International Workshop on Argumentation and Applications (Arg&App 2025)

Oana Cocarascu, Sylvie Doutre, Jean-Guy Mailly, Antonio Rago

Proceedings of the Second International Workshop on Argumentation and Applications (Arg&App 2025)

co-located with the Twenty-Second International Conference on Principles of Knowledge Representation and Reasoning (KR 2025)

Edited By

Oana Cocarascu
Sylvie Doutre
Jean-Guy Mailly
Antonio Rago

King's College London
Université Toulouse Capitole
Imperial College London

November 2025

Preface

In recent years, the increasing availability of data and computational power has driven a remarkable proliferation of Artificial Intelligence (AI) systems in everyday users' lives. However, as data-driven AI systems have become commonplace, it has become increasingly clear that fields from symbolic AI have important roles to play in the future development of these systems, with one such candidate being Computational Argumentation. Formal models of argumentation have received a significant amount of attention in recent years, both within the Knowledge Representation and Reasoning community and from AI researchers in general. Given that argumentation is a mature discipline, it provides not only a wealth of theoretical formalisms suitable for a wide range of tasks, but a whole host of software instantiating these formalisms for real-world settings. These strengths mean that argumentation is particularly adaptable to various application domains, e.g. cyber-democracy, explainable AI, law, medicine, multi-agent systems, public policy making, sustainable development, etc. The goal of this workshop is to emphasise the efforts of the community in this spirit and strengthen the links between formal works on argumentation, their implementations and these domains of application.

The workshop received 3 submissions that were all accepted for presentation during the workshop. They cover a range of topics from the formal foundations of argumentation when deployed in a particular context, to demonstrations of application-driven, argumentative systems. The proceedings also include an abstract of the keynote talk given by Beishui Liao as well as an invited paper describing the ICCMA 2025 competition. We hope that the works presented in the proceedings appeal not only to the growing argumentation community, but also to researchers in general who intend to use computational argumentation in their own applications.

We thank all the authors, the invited speakers Beishui Liao and Johannes Peter Wallner, as well as the program committee members (listed below), for their valued contributions to the workshop.

November 2025

Oana Cocarascu
Sylvie Doutre
Jean-Guy Mailly
Antonio Rago

Program Committee

- Gianvincenzo Alfano, D.I.M.E.S Department, University of Calabria, Italy
- Leila Amgoud, IRIT – CNRS, France
- Katie Atkinson, University of Liverpool, UK
- Hamed Ayoobi, Imperial College London, UK
- Floris Bex, Utrecht University, the Netherlands
- Victor David, INRIA, France
- Jesse Heyninck, Open Universiteit, the Netherlands
- Loan Ho, Vrije Universiteit Amsterdam, the Netherlands
- Anthony Hunter, University College London, UK
- Antonis Kakas, University of Cyprus, Cyprus
- Beishui Liao, Zhejiang University, China
- Xiaolong Liu, Department of Philosophy, Xiangtan University, Xiangtan, China
- Fabio Paglieri, ISTC-CNR Rome, Italy
- Simon Parsons, University of Lincoln, UK
- Guilherme Paulino-Passos, Imperial College London, UK
- Nico Potyka, Cardiff University, UK
- Odinaldo Rodrigues, King's College London, UK
- Ramon Ruiz-Dolz, University of Dundee, UK
- Fabrizio Russo, Imperial College London, UK
- Guillermo R. Simari, Universidad del Sur in Bahia Blanca, Argentina
- Kenneth Skiba, FernUniversität in Hagen, Germany
- Matthias Thimm, FernUniversität in Hagen, Germany
- Francesca Toni, Imperial College London, UK
- Srdjan Vesic, CRIL, CNRS – Univ. Artois, France
- Madeleine Waller, King's College London, UK
- Andreas Xydis, University of Lincoln, UK
- Xiang Yin, Imperial College London, UK
- Liuwen Yu, Luxembourg University, Luxembourg
- Jinfeng Zhong, Université Paris Dauphine, LAMSADE, France

Contents

Argumentation Enabled Explainable AI

Beishui Liao[1,†]

[1]Zhejiang University, China

This talk advances argumentation-enabled explainable AI (XAI) through abstract, structured, and quantitative frameworks. Foundational theories include explanation semantics and attack-defense semantics, formalizing justification-centric reasoning. Structured argumentation integrates value-based norms, while quantitative methods combine human-level knowledge and the implicit knowledge from data, as in e-commerce fraud detection and social media misinformation mitigation. Three approaches are highlighted: (1) rule-based justification combining inductive logic programming and assumption-based argumentation for high-risk decision models; (2) confidence-aware machine learning enhanced by quantitative argumentation, exemplified in an Automated eXplainable Decision System for fake news detection; (3) LLM-driven defeasible reasoning using Chain-of-Thought prompting to align large language models with formal argumentation for transparent inferences. Implementations like the Jiminy Advisor for moral stakeholder agreements and value-driven agents demonstrate ethical and operational solutions. By merging argumentation with AI techniques, we achieve interpretability in fraud prevention, ethical governance, and misinformation mitigation. The talk concludes with insights on balancing computational rigor and societal accountability in complex AI systems.

Arg&App 2025: International Workshop on Argumentation and Applications, November 2025, Melbourne, Australia
baiseliao@zju.edu.cn (B. Liao)

Sixth International Competition on Computational Models of Argumentation: Preliminary Report

Iosif Apostolakis, Andrei Popescu and Johannes P. Wallner

Graz University of Technology, Austria

Abstract

Following the success of previous editions, we present the Sixth International Competition on Computational Models of Argumentation (ICCMA'25). We outline the tracks and rules of the competition, list the benchmarks and participating solvers, and present novelties of this edition.

1. Introduction

Computational argumentation is a research branch within Artificial Intelligence (AI) focused on studying representational and computational aspects that arise in argumentative reasoning [2, 22], with several application avenues, e.g., in medical or legal reasoning [1].

Automated argumentative reasoning is among the central aims in computational argumentation. Based on formalizations of argumentation, such as the prominent argumentation frameworks (AFs) [16], argumentative reasoning is, similar to many symbolic reasoning tasks, in many cases NP-hard [20].

To address this challenge, various algorithmic approaches were studied in the field [11]. To reach maturity of argumentation solvers, i.e., software tools that solve challenging reasoning tasks, a competition, called the International Competition on Computational Models of Argumentation (ICCMA), was introduced in 2015 [33] and then held every two years: in 2017 [23], in 2019 [4, 5], in 2021 [27, 5], and in 2023 [26]. Competitions support development of solvers in that they provide (i) a common platform to submit solvers and evaluate them against each other and (ii) by providing a source of interesting benchmark instances.

In this paper we present the Sixth ICCMA in 2025, or ICCMA'25 for short[1]. Following the success of previous editions, the tasks to be solved are categorized into tracks. Three tracks are on AFs: the Main Track, the Heuristics Track, and the Dynamics Track, concerned with AF reasoning tasks in general, focused on inexact solvers, and dynamically changing AFs, respectively. The ABA Track concerns reasoning tasks in assumption-based argumentation [6]. These tracks are further subdivided into modes and argumentation semantics.

Novel to ICCMA'25 is the inclusion of a fixed SAT solver comparison, in which each solver can participate. Based on observations from previous editions, e.g., [26], SAT solvers play a major role in the algorithmic approaches to NP-hard reasoning tasks in argumentation. Yet, when comparing different argumentation solvers utilizing different SAT solvers, the difference in runtime performance might be mainly due to the *choice of the SAT solver* and not to the algorithmic differences "outside" the SAT solver. By participating in this comparison the solvers are using the same SAT solver, chosen to be CaDiCaL [3] in this edition, and a more direct comparison can be performed.

In this paper we present main ingredients of ICCMA'25. After a brief background on the reasoning tasks we present the tracks, the rules, the benchmark instances, the participating solvers, and a brief discussion of novelties in 2025.

Arg&App 2025: International Workshop on Argumentation and Applications, November 2025, Melbourne, Australia
iosif.apostolakis@tugraz.at (I. Apostolakis); andrei.popescu@tugraz.at (A. Popescu); johannes.p.wallner@tugraz.at (J. P. Wallner)
0009-0003-2124-3773 (I. Apostolakis); 0000-0002-6601-5454 (A. Popescu); 0000-0002-3051-1966 (J. P. Wallner)

[1] More details are available at https://argumentationcompetition.org/2025/index.html

2. Background

We briefly recall the formal background for the reasoning tasks in the tracks of ICCMA'25.

Argumentation Frameworks An argumentation framework (AF) [16] represents abstract arguments and attacks between arguments as a directed graph.

Definition 1. *An argumentation framework (AF) is a pair $F = (A, R)$ with A a finite set of arguments and $R \subseteq A \times A$ a binary relation on A.*

We recall argumentation semantics for AFs next.

Definition 2. *Let $F = (A, R)$ be an AF. A subset $S \subseteq A$ is called conflict-free (in F) if there is no $(a, b) \in R$ with $a, b \in S$. We denote the set of all conflict-free sets of F by cf(F).*

We say that a subset $S \subseteq A$ of arguments defends an argument $a \in A$ (in F) if it holds that for each $(b, a) \in R$ there exists $c \in S$ with $(c, b) \in R$.

For $S \subseteq A$ we define $S_R^+ = \{a \in A \mid (b, a) \in R, b \in S\}$.

Definition 3. *Let $F = (A, R)$ be an AF. Define the characteristic function of F, for $S \subseteq A$, by $\mathcal{F}_F(S) = \{a \mid a$ defended by $S\}$.*

Definition 4. *Let $F = (A, R)$ be an AF, we say that an $E \in$ cf(F) is*

- *an admissible set in F iff $E \subseteq \mathcal{F}_F(E)$,*
- *a complete extension in F iff $E = \mathcal{F}_F(E)$,*
- *a preferred extension in F iff E is an admissible set and there is no admissible set E' in F with $E \subsetneq E'$,*
- *the grounded extension in F iff E is the least fixed-point of $\mathcal{F}_F$,*
- *a stable extension in F iff E attacks each $a \in A \setminus E$,*
- *a semi-stable extension in F iff E is an admissible set in F and there is no admissible set E' in F with $E_R^+ \subsetneq E_R'^+$, and*
- *a stage extension in F iff there is no $E' \in$ cf(F) with $E_R^+ \subsetneq E_R'^+$.*

Reasoning tasks then are defined as follows.

Definition 5. *Let $F = (A, R)$ be an AF, $a \in A$ an argument, and σ a semantics. We say that*

- *a is credulously accepted (in F) under σ if $a \in E$ for some $E \in \sigma(F)$, and*
- *a is skeptically accepted (in F) under σ if $a \in E$ for all $E \in \sigma(F)$.*

Definition 6. *Let $F = (A, R)$ be an AF. Define $S \in$ idl(F) if $S \in$ adm(F) and S contains only arguments skeptically accepted under preferred semantics and there is no $T \in$ adm(F) with $S \subset T$ s.t. T contains only arguments skeptically accepted under preferred semantics.*

The verification task refers to checking whether $S \subseteq A$ is a σ-extension in a given AF. Computational complexity of argumentative reasoning [12, 14, 16, 17, 18, 19, 20, 21] on AFs is summarized in Table 1.

Assumption-based Argumentation We assume a given deductive system $(\mathcal{L}, \mathcal{R})$, where $\mathcal{L}$ is a set of atoms, and $\mathcal{R}$ a set of inference rules over $\mathcal{L}$. A rule $r \in \mathcal{R}$ has the form $a_0 \leftarrow a_1, \ldots, a_n$ with $a_i \in \mathcal{L}$. We denote the head of rule r by $head(r) = a_0$ and the (possibly empty) body of r with $body(r) = \{a_1, \ldots, a_n\}$.

An atom a is derivable from a set of assumptions $A \subseteq \mathcal{A}$, and rules $\mathcal{R}$, if $a \in A$ or there is a sequence of rules $(r_1, \ldots, r_n)$, $r_i \in \mathcal{R}$, such that $head(r_n) = a$ and for each rule in the sequence we find that the body of this rule is fully contained in A and heads of previous rules in the sequence (for each i, $1 \leq i \leq n$, we find $body(r_i) \subseteq A \cup \{head(r_j) \mid j < i\}$). We say A derives a for short.

[2] Under randomized reductions completeness results were obtained [19].

Table 1
Computational complexity of AF reasoning

σ	credulous acceptance	skeptical acceptance	verification
adm	NP-complete	trivial	in P
com	NP-complete	in P	in P
idl[2]	in Θ_2^P	in Θ_2^P	in Θ_2^P
prf	NP-complete	Π_2^P-complete	coNP-complete
stb	NP-complete	coNP-complete	in P
sem	Σ_2^P-complete	Π_2^P-complete	coNP-complete

Table 2
Complexity of deciding acceptance of an atom in flat ABA.

σ	credulous acceptance	skeptical acceptance	verification
adm	NP-complete	in P	in P
com	NP-complete	in P	in P
prf	NP-complete	Π_2^P-complete	coNP-complete
stb	NP-complete	coNP-complete	in P

Definition 7. *An ABA framework is a tuple $F = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^-)$, where $(\mathcal{L}, \mathcal{R})$ is a deductive system, $\mathcal{A} \subseteq \mathcal{L}$ a non-empty set of assumptions, and $^-$ a function mapping assumptions $\mathcal{A}$ to atoms $\mathcal{L}$.*

We restrict our attention to so-called flat ABA frameworks, which can be represented as ABA frameworks where heads of rules contain no assumption.

Definition 8. *Let $F = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^-)$ be an ABA framework, and $A, B \subseteq \mathcal{A}$ be two sets of assumptions. Assumption set A attacks assumption set B in F if A' derives $\overline{b}$ for some $A' \subseteq A$ and $b \in B$.*

Similar to notions of abstract argumentation (such as AFs), conflict-freeness and defense can be defined on ABA frameworks, as follows.

Definition 9. *Let $F = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^-)$ be an ABA framework. An assumption set $A \subseteq \mathcal{A}$ is conflict-free in F iff A does not attack itself. Set A defends assumption set $B \subseteq \mathcal{A}$ in F iff for all $C \subseteq \mathcal{A}$ that attack B it holds that A attacks C.*

Definition 10. *Let $F = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^-)$ be an ABA framework. Further, let $A \subseteq \mathcal{A}$ be a conflict-free set of assumptions in F. In F, set A is*

- admissible *iff A defends itself;*
- complete *iff A is admissible and contains every assumption set defended by A;*
- preferred *iff A is admissible and there is no admissible set of assumptions B with $A \subset B$; and*
- stable *iff each $\{x\} \subseteq \mathcal{A} \setminus A$ is attacked by A.*

Credulous and skeptical acceptance are defined analogously as in AFs, except that one asks for credulous or skeptical acceptance of an atom $x \in \mathcal{L}$, under a specified semantics σ. Complexity of reasoning in flat ABA frameworks [15, 13] is summarized in Table 2.

3. Tracks of the Competition

In the 2025 edition of ICCMA, the established *main track*, *heuristics track* (formerly the approximate track), *dynamic track*, and *ABA track* are included. In brief, the main, heuristics, and dynamic tracks concern AF reasoning tasks, while the ABA track is focused on ABA reasoning tasks. Each track consists of several sub tracks, determined by a combination of a semantics and a mode of reasoning.

Reasoning Tasks We consider the following modes:

- decide credulous acceptance (DC),
- decide skeptical acceptance (DS), and
- return some extension (SE).

The term "extension" is here used slightly ambiguously: we call both σ-extensions in AFs and σ-sets in ABAs "extensions" (mainly for the sake of a uniform presentation).

Each of these modes are combined with semantics: complete (CO), preferred (PR), stable (ST), semi-stable (SST), and ideal (ID). Then a reasoning task has the form XX-YY with XX the mode and YY a semantics, e.g., DS-PR stands for decide skeptical acceptance under preferred semantics.

All reasoning tasks considered in ICCMA'25 are NP-hard or coNP-hard (for the decision problems) or a task for finding an extension under a semantics with NP-hard or coNP-hard decision tasks.

Main Track In this track sequential core solvers in open-source for reasoning tasks are evaluated. In contrast, portfolio solvers, multi-processor systems, and tools not available in open-source could be submitted to the No-Limits Track, which is the same as the main track, except for the aforementioned differences. Ranking is based on CPU time, except for the No-Limits Track, where wall clock time is used. The concrete sub tracks are DC-{CO|ST|SST}, DS-{PR|ST|SST}, and SE-{PR|ST|SST|ID}. For the DC and DS modes, the expected answers consist of "YES" and "NO", and in specified cases an associated witness, see Section 4.

Heuristics Track In essence similar to the Main Track, but shorter timeouts are enforced, and no witness is required. Since the focus is on inexact solvers, incorrect solutions are discarded. The sub tracks are DC-{CO|ST|SST|ID} and DS-{PR|ST|SST}.

Dynamic Track In this track, as in the previous edition of ICCMA [26], solvers are called via an interface, called IPAFAIR, which solvers need to implement. This allows the addition and removal of parts of an AF, and asking the solver to decide reasoning tasks. The sub tracks are DC-CO, DS-PR, DC-ST, and DS-ST.

ABA Track This track follows the same basis as the main track, except that ABA frameworks and associated reasoning tasks are considered. The sub tracks are DC-{CO|ST}, DS-{PR|ST}, and SE-{PR|ST}.

Fixed SAT-Solver Comparison New in this edition is the possibility to submit a solver to be part of a special comparison where the used SAT solver is fixed by the organizers of the competition. This is achieved by requiring that participating solvers implement the IPASIR interface for (incremental) SAT solvers[3]. In this way, the SAT solver is called via an interface, and the underlying SAT solver can be replaced. We chose CaDiCaL [3], version 2.1.3, as the fixed SAT solver.

4. Rules

We briefly outline the rules of the competition, and otherwise refer the interested reader to the competition webpage.

[3] https://github.com/biotomas/ipasir

Correctness A solver is not immediately disqualified in case of incorrect results. We allow solver developers time to fix bugs in their solvers.

Correctness requires, in addition to having the correct output syntax, an answer returned on the command line, depending on the track. In the Heuristics Track incorrect solutions are simply discarded (as here the focus is on quick answers). For the Dynamic Track, the solver is called via an interface and a correct implementation of the interface is required. In addition to returning "YES" or "NO", a witness can be required for the Main Track and No-Limits Track. In the ABA Track, mainly because solvers for ABA are a relatively recent development, currently no witness is required.

- When deciding credulous acceptance (DC), in case of a "YES" answer, a witness must be provided.
  - For the case of DC-CO, the witness consists of an admissible set containing the queried argument.
  - For other cases, i.e., DC-YY, for a semantics YY, we require a witness extension of the semantics YY containing the queried argument.
- When deciding skeptical acceptance (DS), in case of a "NO" answer, a witness must be provided.
  - For the case of DS-PR, the witness can either consist of a preferred extension not containing the queried argument or, alternatively, of an admissible set attacking the queried argument.
  - For all other cases, a witness consists of an extension under the specified semantics not containing the queried argument.

With the exception of preferred and semi-stable semantics, witnesses can be verified in polynomial time, for DS-PR if an admissible set attacking the queried argument is returned, this can also be checked in polynomial time. Checking whether a given set of arguments is a preferred or a semi-stable extension is in general coNP-complete.

Open Source Requirement and Solver Description Except for submissions to the No-Limit Track, all solvers must provide an open-source implementation, which will be published after final announcements of the results. Moreover, a description must be provided.

Compilation The solvers must be compilable using standard Unix tools.

Input Format We followed the input format specifications of ICCMA'23, which specify compact representations of AFs and ABAs, in a DIMACS-like format.

Ranking and Limits Per instance, a limit of 1200 seconds CPU time was enforced, except for the No-Limits Track where we enforced 1200 wall clock time, and for the Heuristics Track where we enforced 60 seconds CPU time. For ranking PAR-2 scoring is used. That is, the score for a given solver on an instance is $2 \cdot 1200$ if the solver timed out on this instance, and otherwise the CPU running time of the solver on this instance in seconds. The score for a solver on a subtrack is the sum of the scores of this solver over all instances of the subtrack. The winner is the solver with the lowest score. For the Heuristics Track, the number of correct solutions decides the ranking.

5. Benchmarks

Argumentation Frameworks We considered all previous benchmark instances and two novel submitted benchmarks. That is, we included the generators or benchmark suites called GroundedGenerator, SCCGenerator and the StableGenerator [9], ABA2AF [29], AdmBuster [8], a generator [10] for AF families called Barabasi-Albert, Erdos-Renyi and Watts-Strogatz, Planning2AF [34, 32], Sembuster, Traffic, AFGen [25], Datalog [35], and crusti_g2io [28]. New for this edition are KWT by Kuhlmann and Thimm, and AFSubsampling by Lars Malmqvist. The former aims to generate AFs that are challenging

to solve under the DS-PR reasoning task, by carefully crafting such instances. The latter generates AFs from existing ones by systematic subsampling.

We sampled up to 20 AFs from existing benchmarks, generating uniformly at random query arguments, except for self-attacking arguments and arguments without incoming attacks (to avoid some trivial cases). We also excluded AFs with fewer than 100 arguments. For the two novel benchmarks, we considered up to 30 AFs. Overall, this resulted in 322 AFs.

ABA Frameworks We included the RandomABAGenerator from the previous edition, and two novel ABA generators: "ABAs from the Wild" by Odekerken and ABCGen by Niskanen, Rankooh, Lehtonen, and Järvisalo. The former generates ABA frameworks (and AFs) from ASPIC$^+$ theories [30] similar in structure to theories used in a real-world application at the Dutch National Police [31]. The latter generator uses a clustered approach, inspired by the StableGenerator for AFs, with the aim of generating challenging ABA frameworks. After preliminary testing, we decided to utilize the former generator only for the ABA Track.

We considered 80 instances from the previous generator and 120 each for the new generators, for 320 ABA instances. The atoms to query are selected by sampling uniformly from the set of atoms that are (i) derivable from non-self contrary assumptions and (ii) not derivable by considering only assumptions without any contrary (to again avoid some trivial cases).

6. Submitted Solvers

We list the submitted solvers to ICCMA'25 in alphabetical order.

100BA by Niskanen, Rankooh, Lehtonen, and Järvisalo is a solver based on SAT solving utilizing non-trivial SAT encodings and propagation methods for ABA frameworks.

Acbar by Lehtonen, Rapberger, and Ulbricht is a solver for ABA frameworks that translates a given instance to an AF and using mu-toksia for solving the AF.

AFCA version 0.1, submitted by Obiedkov and Sertkaya, utilizes algorithms developed in the area of formal concept analysis [24].

ARIPOTER version 2, submitted by Cibier, Delobelle, Mailly, and Rossit, is a solver in the heuristics track based on utilizing concepts regarding the grounded semantics.

ASPforABA submitted by Lehtonen, is a solver for ABA frameworks based on answer set programming (ASP) [7].

ASSAT solver family submitted by Liu et al. consisting of four solvers utilizing optimizations of strategies of SAT solvers.

fargo-limited version 2.2.2, submitted by Thimm, utilizes a DPLL-based approach with a bounded depth or bounded number of sub queries to heuristically solve reasoning tasks in AFs.

fargo-timelimited by Cibier and Mailly adapts fargo-limited combined with a lightweight neural network for the Heuristics Track.

FastAFGCN version 0.1, submitted by Malmqvist, uses insights from the grounded semantics and graph neural networks for solving tasks in the Heuristics Track.

FastLiGS submitted by Cibier and Mailly uses neuronal networks for the Heuristics Track.

Fudge version 3.3.3, submitted by Cerutti, Vallati, and Thimm, utilizes a SAT-based approach with novel encodings.

harper++ version 1.1.2, submitted by Thimm utilizes the grounded semantics for heuristical AF reasoning.

Heuback version 0.1, submitted by Malmqvist uses the grounded semantics, strongly connected components, and limited backtracking techniques for solving reasoning tasks in the Heuristics Track.

MS-DIS submitted by Diller and Gorczyca, is aimed at dialectical explication of reasoning in ABA frameworks making use of ASP.

mu-toksia version 2025.06.06, submitted by Niskanen, is a SAT-based approach to solving reasoning tasks in AFs.

reducto version 2.121, submitted by Bengel, Sander, and Thimm, is a SAT-based approach that uses a reduct-based characterization for skeptical acceptance under preferred semantics.

Scallop submitted by Lagniez, Lonca, and Mailly utilizes a SAT solver for AF and ABA reasoning using various shortcuts.

SMART version 1.0, submitted by Hoffmann, Kuhlmann, and Thimm uses graph neural networks for AF reasoning.

7. Computational Infrastructure

The competition is run on the scientific computing infrastructure called Austrian Scientific Computing (ASC), a collaboration of several Austrian universities to provide resources and technical support for users. ICCMA'25 is run on the VSC-4 cluster.

8. Novelties in ICCMA 2025

New to this edition of ICCMA are (i) the fixed SAT solver comparison, (ii) simplification of some witnesses, and (iii) the stage semantics was discontinued from the last editions.

The addition of the fixed SAT solver comparison is seen as critical in performance comparison by part of the community, to more clearly assess strengths of algorithms rather than differences of underlying SAT solvers. Nevertheless, SAT solvers appear to play a major role in reasoning in argumentation. This step is mainly to see differences in algorithmic approaches using SAT.

We allow for somewhat simplified witnesses in some cases. For CO (complete semantics) also admissible sets may be returned, since every complete extension is also admissible. Sometimes admissible sets containing a queried argument may be simpler to obtain than complete extensions containing the queried argument. For DS-PR, we also allow admissible sets attacking the queried argument, then it directly follows that there is a preferred extension not containing the queried argument. This "shortcut" allows for finding possibly compact witnesses. Finally, the number of different semantics (and reasoning tasks) was relatively high in previous editions of ICCMA. This is why we decided to focus the choice of sub tracks somewhat.

9. Conclusions

We presented the main components, benchmarks (submissions), and solver submissions of ICCMA'25. We think development of mature solvers can significantly benefit from regular competitions like ICCMA. Both novel solvers and novel benchmark instances can fruitfully extend the reach of argumentation solvers, and argumentation tools in a more general sense.

Acknowledgements

The competition organizers want to thank all developers of solvers and all developers of benchmark instances. These are essential for a lively competition and they strengthen the landscape of argumentation tools for scientific and application use.

The computational results have been achieved using the Austrian Scientific Computing (ASC) infrastructure.

This research was funded in whole or in part by the Austrian Science Fund (FWF) P35632. For open access purposes, the authors have applied a CC BY public copyright license to any author accepted manuscript version arising from this submission.

References

[1] K. Atkinson, P. Baroni, M. Giacomin, A. Hunter, H. Prakken, C. Reed, G. R. Simari, M. Thimm, S. Villata, Towards artificial argumentation, AI Mag. 38 (2017) 25–36.

[2] P. Baroni, D. Gabbay, M. Giacomin, L. van der Torre (Eds.), Handbook of Formal Argumentation, College Publications, 2018.

[3] A. Biere, T. Faller, K. Fazekas, M. Fleury, N. Froleyks, F. Pollitt, CaDiCaL 2.0, in: A. Gurfinkel, V. Ganesh (Eds.), Proc. CAV, volume 14681 of *Lecture Notes in Computer Science*, Springer, 2024, pp. 133–152.

[4] S. Bistarelli, L. Kotthoff, F. Santini, C. Taticchi, Summary report for the third international competition on computational models of argumentation, AI Mag. 42 (2021) 70–73.

[5] S. Bistarelli, L. Kotthoff, J.-M. Lagniez, E. Lonca, J.-G. Mailly, J. Rossit, F. Santini, C. Taticchi, The third and fourth international competitions on computational models of argumentation: Design, results and analysis, Argument Comput. 16 (2025).

[6] A. Bondarenko, P. M. Dung, R. A. Kowalski, F. Toni, An abstract, argumentation-theoretic approach to default reasoning, Artif. Intell. 93 (1997) 63–101.

[7] G. Brewka, T. Eiter, M. Truszczynski, Answer set programming at a glance, Commun. ACM 54 (2011) 92–103.

[8] M. Caminada, P. E. Dunne, Strong admissibility revisited: Theory and applications, Argument Comput. 10 (2019) 277–300.

[9] F. Cerutti, N. Oren, H. Strass, M. Thimm, M. Vallati, A benchmark framework for a computational argumentation competition, in: S. Parsons, N. Oren, C. Reed, F. Cerutti (Eds.), Proc. COMMA, volume 266 of *Frontiers in Artificial Intelligence and Applications*, IOS Press, 2014, pp. 459–460.

[10] F. Cerutti, M. Giacomin, M. Vallati, Generating structured argumentation frameworks: Afbenchgen2, in: P. Baroni, T. F. Gordon, T. Scheffler, M. Stede (Eds.), Proc. COMMA, volume 287 of *Frontiers in Artificial Intelligence and Applications*, IOS Press, 2016, pp. 467–468.

[11] F. Cerutti, S. A. Gaggl, M. Thimm, J. P. Wallner, Foundations of implementations for formal argumentation, in: P. Baroni, D. Gabbay, M. Giacomin, L. van der Torre (Eds.), Handbook of Formal Argumentation, College Publications, 2018, pp. 688–767.

[12] S. Coste-Marquis, C. Devred, P. Marquis, Symmetric argumentation frameworks, in: L. Godo (Ed.), Proc. ECSQARU, volume 3571 of *Lecture Notes in Computer Science*, Springer, 2005, pp. 317–328.

[13] K. Cyras, Q. Heinrich, F. Toni, Computational complexity of flat and generic assumption-based argumentation, with and without probabilities, Artif. Intell. 293 (2021) 103449.

[14] Y. Dimopoulos, A. Torres, Graph theoretical structures in logic programs and default theories, Theor. Comput. Sci. 170 (1996) 209–244.
[15] Y. Dimopoulos, B. Nebel, F. Toni, On the computational complexity of assumption-based argumentation for default reasoning, Artif. Intell. 141 (2002) 57–78.
[16] P. M. Dung, On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games, Artif. Intell. 77 (1995) 321–358.
[17] P. E. Dunne, T. J. M. Bench-Capon, Coherence in finite argument systems, Artif. Intell. 141 (2002) 187–203.
[18] P. E. Dunne, M. Caminada, Computational complexity of semi-stable semantics in abstract argumentation frameworks, in: S. Hölldobler, C. Lutz, H. Wansing (Eds.), Proc. JELIA, volume 5293 of *Lecture Notes in Computer Science*, Springer, 2008, pp. 153–165.
[19] P. E. Dunne, The computational complexity of ideal semantics, Artif. Intell. 173 (2009) 1559–1591.
[20] W. Dvořák, P. E. Dunne, Computational problems in formal argumentation and their complexity, in: P. Baroni, D. Gabbay, M. Giacomin, L. van der Torre (Eds.), Handbook of Formal Argumentation, College Publications, 2018, pp. 631–688.
[21] W. Dvořák, S. Woltran, Complexity of semi-stable and stage semantics in argumentation frameworks, Inf. Process. Lett. 110 (2010) 425–430.
[22] D. Gabbay, M. Giacomin, G. R. Simari, M. Thimm (Eds.), Handbook of Formal Argumentation, volume 2, College Publications, 2021.
[23] S. A. Gaggl, T. Linsbichler, M. Maratea, S. Woltran, Design and results of the second international competition on computational models of argumentation, Artif. Intell. 279 (2020).
[24] B. Ganter, R. Wille, Formal Concept Analysis - Mathematical Foundations, 2 ed., Springer, 2024.
[25] Y. Gao, A random model for argumentation framework: Phase transitions, empirical hardness, and heuristics, in: C. Sierra (Ed.), Proc. IJCAI, ijcai.org, 2017, pp. 503–509.
[26] M. Järvisalo, T. Lehtonen, A. Niskanen, ICCMA 2023: 5th international competition on computational models of argumentation, Artif. Intell. 342 (2025) 104311.
[27] J. Lagniez, E. Lonca, J. Mailly, J. Rossit, Introducing the fourth international competition on computational models of argumentation, in: S. A. Gaggl, M. Thimm, M. Vallati (Eds.), Proc. SAFA, volume 2672 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2020, pp. 80–85.
[28] J. Lagniez, E. Lonca, J. Mailly, J. Rossit, A new evolutive generator for graphs with communities and its application to abstract argumentation, in: Z. Bouraoui, F. Schwarzentruber, A. Wilczynski (Eds.), Proc. JIAF, 2023, pp. 28–38.
[29] T. Lehtonen, J. P. Wallner, M. Järvisalo, From structured to abstract argumentation: Assumption-based acceptance via AF reasoning, in: A. Antonucci, L. Cholvy, O. Papini (Eds.), Proc. ECSQARU, volume 10369 of *Lecture Notes in Computer Science*, Springer, 2017, pp. 57–68.
[30] S. Modgil, H. Prakken, A general account of argumentation with preferences, Artif. Intell. 195 (2013) 361–397.
[31] D. Odekerken, F. Bex, A. Borg, B. Testerink, Approximating stability for applied argument-based inquiry, Intell. Syst. Appl. 16 (2022) 200110.
[32] A. Sideris, Y. Dimopoulos, Constraint propagation in propositional planning, in: R. I. Brafman, H. Geffner, J. Hoffmann, H. A. Kautz (Eds.), Proc. ICAPS, AAAI, 2010, pp. 153–160.
[33] M. Thimm, S. Villata, The first international competition on computational models of argumentation: Results and analysis, Artif. Intell. 252 (2017) 267–294.
[34] A. Z. Wyner, T. J. M. Bench-Capon, P. E. Dunne, F. Cerutti, Senses of 'argument' in instantiated argumentation frameworks, Argument Comput. 6 (2015) 50–72.
[35] B. Yun, M. Croitoru, S. Vesic, P. Bisquert, DAGGER: datalog+/- argumentation graph generator, in: E. André, S. Koenig, M. Dastani, G. Sukthankar (Eds.), Proc. AAMAS, IFAAMAS, 2018, pp. 1841–1843.

Heterogeneous Graph Neural Networks for Credulous Acceptance of Assumptions in ABA

Preesha Gehlot, Anna Rapberger*, Fabrizio Russo* and Francesca Toni

Imperial College London, Department of Computing

Abstract

Assumption-Based Argumentation (ABA) is a powerful structured argumentation formalism, but exact computation of extensions under stable semantics is intractable for large frameworks. We present the first Graph Neural Network (GNN) approach to approximate credulous acceptance in ABA. To use GNNs, we represent ABA frameworks via a dependency graph representation that encodes atoms and rules as nodes and distinguishes support, derive and attack relations by heterogeneous edge labels. We propose two GNN variants—ABAGCN and ABAGAT—that stack residual heterogeneous convolution or attention blocks, respectively, to learn node embeddings. Our models are trained on the ICCMA2023 benchmark, augmented with synthetic ABAFs, with hyperparameters optimised via Bayesian search. Empirically, both ABAGCN and ABAGAT outperform a state-of-the-art GNN baseline that we adapt from the abstract argumentation literature, achieving an F1 score up to 0.71 on the ICCMA instances. Finally, we develop a poly-time extension-reconstruction algorithm driven by our predictor: it reconstructs stable extensions with F1 above 0.85 on small ABAFs and maintains an F1 of about 0.58 on frameworks with 1,000 atoms. Our work opens new avenues for scalable approximate reasoning in structured argumentation.

Keywords

Assumption-Based Argumentation, Approximate Reasoning, Graph Neural Networks

1. Introduction

Computational argumentation provides formal tools for modelling defeasible reasoning over conflicting information. In the *abstract* setting, arguments are treated as opaque nodes and attacks as binary relations, giving rise to abstract argumentation frameworks (AFs) [1]. By contrast, *structured* formalisms expose the internal composition of arguments. Assumption-Based Argumentation (ABA) is a prominent structured framework, enabling transparent reasoning in domains such as decision support [2], planning [3], and causal discovery [4].

The building blocks of ABA are *assumptions*, which are the defeasible elements of an ABA framework (ABAF), and *inference rules*, which are used to construct *arguments* based on a given set of assumptions. *Conflicts* arise between assumption sets S and T if the conclusion of an argument constructed from S is the contrary of some assumption in T; we say that S attacks T in this case. *Argumentation semantics* provide criteria which renders sets of assumptions as jointly acceptable: the so-called *extensions*. One of the most popular semantics is the stable semantics which accepts a set of assumptions S only if it has no internal conflicts and attacks every set not contained in S. While stable semantics neatly characterise acceptable assumption sets, they are hard to compute: membership testing for stable extensions is NP-complete [5]. Since real-world ABA frameworks can be very large [4], relying on exact approaches is often not viable.

In this work, we address this issue by using Graph Neural Networks (GNNs) [6] to predict the acceptability of assumptions. GNNs leverage the relationships between nodes and edges in a graph to learn representations that capture the its structure and allow to predict labels for nodes, edges or the whole graph. This specialised architecture allows GNNs to *amortise* the graph analysis and use its stored weights to predict feature of interest (e.g. node labels as in our case) in constant time after the initial network training, thus introducing significant efficiency at inference time.

Arg&App 2025: International Workshop on Argumentation and Applications, November 2025, Melbourne, Australia

*Corresponding authors

a.rapberger@imperial.ac.uk (A. Rapberger); fabrizio@imperial.ac.uk (F. Russo)

While GNNs have been successfully used to approximate the acceptability of AFs, see, e.g., [7, 8, 9], their potential remains unexplored for ABA and any other structured argumentation formalism to date. In this work, we present the first GNN architecture to predict credulous acceptance for ABA under stable semantics. Our contributions are as follows:

- We introduce a faithful *dependency graph* encoding of ABAFs that distinguishes assumption, claim and rule nodes as well as support, derive, and attack edges.
- We develop two heterogeneous GNN architectures—ABAGCN and ABAGAT—composed of residual stacks of relation-specific convolutional or attention layers, respectively, enriched with learnable embeddings and degree-based features.
- We implement a training and evaluation pipeline combining the ICCMA2023 ABA benchmarks with 19,500 synthetically generated ABAFs. Our experiments demonstrate that both ABAGCN and ABAGAT outperform an AF-based GNN baseline based on state-of-the-art AFGCNv2, achieving node-level F1 up to 0.71 on small ABAFs and 0.65 overall.
- Driven by our credulous acceptance predictor, we develop a polynomial-time extension-reconstruction algorithm which reconstructs stable extensions with high fidelity. Our algorithm attains F1>0.85 on small frameworks and remains above 0.58 on ABAFs with 1,000 atoms while running 2.3× faster than a state-of-the-art exact solver for the biggest (4,000 and 5,000 atoms) and most challenging instances.

2. Related work

Kuhlmann and Thimm [7] were the first to frame credulous acceptance in an AF as a graph-classification problem. They generated random AFs, labeled each argument's acceptability using the exact CoQuiAAS solver [10], and trained a Graph Convolutional Network (GCN) on two feature sets: the raw adjacency matrix and adjacency augmented with each node's in- and out-degree. Including degree information improved accuracy to $\sim$80%, with F1 scores below 0.4, and class-balancing during training enhanced detection of under-represented arguments. Despite only marginal effects from graph topology or dataset size, their GCN reduced runtime from over an hour (CoQuiAAS) to under 0.5s for a full test set.

Malmqvist et al. [9] then proposed AFGCN, which augments node features with 64-dimensional DeepWalk embeddings [11] before feeding them—alongside the AF's adjacency matrix—through 4–6 residual GCN–dropout blocks. To address the skewed accept/reject ratio in real AFs, they exclude frameworks with very few accepted arguments and introduce a randomised per-epoch masking of labels, forcing the model to infer hidden acceptabilities. This scheme, rather than network depth or explicit class-balancing, accounts for most of AFGCN's increase to $\sim$82% accuracy on rebalanced data (62% reported for [7] on the same data). Malmqvist et al. [12] enrich argument representations with graph-level metrics (centralities, PageRank, colouring) and use four GCN–ReLU–dropout blocks plus an epoch-wise reshuffle-and-rebalance routine and a pre-check on the grounded extension for faster and more accurate inference.

Cibier and Mailly [8] optimise AFGCNv2 by reimplementing AF parsing and graph-metric computation in Rust, slashing preprocessing time and memory usage. They evaluate the original AFGCNv2 against two variants—adding five semantic features to random features, or retaining only 11 "meaningful" features—and assess each with and without dropout. Their results show that pruning degrades accuracy, and that the no-dropout version of "feature-enhanced" version achieves the best overall performance ($\sim$83% accuracy vs $\sim$75% reported for AFGCNv2 on the same data), highlighting the importance of rich semantic features and careful dropout placement. Finally, they introduce AFGAT, a three-layer GATv2 (an updated attention mechanism [13] with multi-head attention), which surpasses all GCN variants obtaining $\sim$87% on the same benchmark data from ICCMA.

Craandijk and Bex [14] propose AGNN, which learns argument embeddings through iterative message passing that internalises conflict-freeness and defense, then uses the resulting acceptance probabilities to drive a constructive backtracking algorithm for extension enumeration. We build on this enumeration strategy in our experiments in Section 5.

We adopt AFGCNv2 [12] as our baseline[1] but, inspired by the results in [8], we replace its handcrafted features with learnable embeddings and experiment with both GCNs and GAT layers in our final models. Additionally, we handle class imbalance via loss weighting rather than mini-batch rebalancing and disable the grounded-extension heuristic to test the models' predictive accuracy on the entire frameworks rather than on the non-grounded portion of it only. In contrast to the AF-focused solvers discussed, we present the first native approximate solver for ABAFs, combining an enriched and faithful graphical representation with heterogeneous GCN and GAT-based prediction of assumption acceptability incorporated in an AGNN-inspired extension reconstruction.

3. Preliminaries

We recall the relevant elements of abstract and assumption-based argumentation, as well as GNNs.

3.1. Abstract Argumentation

An argumentation framework (AF) [1] is a directed graph $F = (\mathsf{A}, \mathsf{R})$ where A are arguments and $\mathsf{R} \subseteq \mathsf{A} \times \mathsf{A}$ are *attacks*. For $x, y \in \mathsf{A}$, if $(x, y) \in \mathsf{R}$ we say x *attacks* y; $E \subseteq \mathsf{A}$ *defends* x if it attacks each attacker of x; E is *conflict-free* ($E \in cf(F)$) iff it does not attack itself. A semantics σ is a function that assigns each AF a set of sets of arguments, so-called *extensions*. Here, we focus on stable semantics.

Definition 3.1. *Let $F = (\mathsf{A}, \mathsf{R})$ be an AF. A set $E \subseteq \mathsf{A}$ is* stable *($E \in stb(F)$) iff E is conflict-free and attacks each x for every $x \in \mathsf{A} \setminus E$.*

3.2. Assumption-based Argumentation

We assume a *deductive system*, i.e. a tuple $(\mathcal{L}, \mathcal{R})$, where $\mathcal{L}$ is a set of atoms and $\mathcal{R}$ is a set of inference rules over $\mathcal{L}$. A rule $r \in \mathcal{R}$ has the form $a_0 \leftarrow a_1, \ldots, a_n$, s.t. $a_i \in \mathcal{L}$ for all $0 \leq i \leq n$; $head(r) := a_0$ is the *head* and $body(r) := \{a_1, \ldots, a_n\}$ is the (possibly empty) *body* of r.

Definition 3.2. *An ABA framework (ABAF) [15] is a tuple $(\mathcal{L}, \mathcal{R}, \mathcal{A}, {}^-)$, where $(\mathcal{L}, \mathcal{R})$ is a deductive system, $\mathcal{A} \subseteq \mathcal{L}$ a set of assumptions, and ${}^- : \mathcal{A} \to \mathcal{L}$ a contrary function.*

We fix an arbitrary ABAF $\mathcal{D} = (\mathcal{L}, \mathcal{R}, \mathcal{A}, {}^-)$ below. The ABAF $\mathcal{D}$ is *flat* iff $head(r) \notin \mathcal{A}$ for all $r \in \mathcal{R}$. In this work we focus on *flat* and *finite* ABAFs, i.e. $\mathcal{L}$ and $\mathcal{R}$ are finite.

An atom $p \in \mathcal{L}$ is tree-derivable from assumptions $S \subseteq \mathcal{A}$ and rules $R \subseteq \mathcal{R}$, denoted by $S \vdash_R p$, if there is a finite rooted labeled tree t s.t. i) the root of t is labeled with p, ii) the set of labels for the leaves of t is equal to S or $S \cup \{\top\}$, and iii) for each node v that is not a leaf of t there is a rule $r \in R$ such that v is labeled with $head(r)$ and labels of the children correspond to $body(r)$ or $\top$ if $body(r) = \emptyset$. We write $S \vdash p$ iff there exists $R \subseteq \mathcal{R}$ such that $S \vdash_R p$.

Let $S \subseteq \mathcal{A}$. By $\overline{S} := \{\overline{a} \mid a \in S\}$ we denote the set of all contraries of S. The set S *attacks* a set $T \subseteq \mathcal{A}$ if there are $S' \subseteq S$ and $a \in T$ s.t. $S' \vdash \overline{a}$; if S attacks $\{a\}$ we say S attacks a. S is conflict-free ($S \in cf(\mathcal{D})$) if it does not attack itself. We define stable ABA semantics.

Definition 3.3. *Let $\mathcal{D} = (\mathcal{L}, \mathcal{R}, \mathcal{A}, {}^-)$ be an ABAF. A set $S \subseteq \mathcal{A}$ is* stable *($S \in stb(\mathcal{D})$) iff S is conflict-free and attacks each $\{x\}$ for every $x \in \mathcal{A} \setminus S$.*

As for AFs, we call a set $S \in stb(\mathcal{D})$ an *extension*. An assumption $a \in \mathcal{A}$ is *credulously accepted* w.r.t. stb in an ABAF $\mathcal{D}$ iff there is some $S \in stb(\mathcal{D})$ with $a \in S$.

Example 3.4. *Consider an ABAF $\mathcal{D} = (\mathcal{L}, \mathcal{R}, \mathcal{A}, {}^-)$ with atoms $\mathcal{L} = \{a, b, c, d, p, \overline{a}, \overline{b}, \overline{c}, \overline{d}\}$, assumptions $\mathcal{A} = \{a, b, c, d\}$, their contraries $\overline{a}$, $\overline{b}$, $\overline{c}$, $\overline{d}$, respectively, and rules $r_1, \ldots, r_4$, respectively:*

$$\overline{c} \leftarrow a, d \qquad p \leftarrow b \qquad \overline{d} \leftarrow c \qquad \overline{a} \leftarrow p, c$$

[1] Available at https://github.com/lmlearning/AFGraphLib/tree/main/AFGCNv2. We could not use the improved versions from [8] as no weights were made available.

We obtain two stable sets here: $\{b, c\}$ *and* $\{a, b, d\}$*. Indeed,* $\{b, c\}$ *attacks* a *and* d *since* $\{b, c\} \vdash \overline{a}$ *and* $\{c\} \vdash \overline{d}$*; and* $\{a, b, d\}$ *attacks* c *since* $\{a, d\} \vdash \overline{c}$*.*

Viewing tree derivations as arguments, an ABAF induces an AF as follows [16, 17].

Definition 3.5. *The associated AF* $F_{\mathcal{D}} = (\mathsf{A}, \mathsf{R})$ *of an ABA* $\mathcal{D} = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^{-})$ *is given by* $\mathsf{A} = \{S \vdash p \mid \exists R \subseteq \mathcal{R} : S \vdash_R p\}$ *and attack relation* $(S \vdash p, S' \vdash p') \in \mathsf{R}$ *iff* $p \in \overline{S'}$*.*

Semantics of AFs and ABAFs are closely related [16].

Proposition 3.6. *Given an ABA* $\mathcal{D} = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^{-})$ *and its corresponding AF* $F = (\mathsf{A}, \mathsf{R})$*. If* $E \in stb(F)$ *then* $\bigcup_{S \vdash p \in E} S \in stb(\mathcal{D})$*; if* $S \in stb(\mathcal{D})$ *then* $\{S' \vdash p \mid \exists S' \subseteq S, R \subseteq \mathcal{R} : S' \vdash_R p\} \in \sigma(F)$*.*

Note that the AF instantiation can be exponential in the size of the given ABAF $\mathcal{D}$. To tackle this issue, Lehtonen et al. [18] propose a poly-time preprocessing technique that yields an ABAF so that the resulting AF is polynomial in the size of $\mathcal{D}$. Their procedure handles the derivation circularity and flattens out the nested argument construction; they show that this construction preserves semantics under projection. We refer the interested reader to [18] for more details on the construction.

3.3. Graph Neural Networks

A neural network (NN) is a parametrised function that maps an input vector $\mathbf{x} \in \mathbb{R}^n$ to an output through successive layers of neurons. We set the input neurons as $\mathbf{h}^{(0)} = \mathbf{x}$ and denoting by $\mathbf{h}^{(l)}$ the row-vector of neurons at layer l, each neuron at layer l computes

$$h_i^{(l+1)} \;=\; f\big((\mathbf{w}_i^{(l)})^\top \mathbf{h}^{(l)} + b_i^{(l)}\big),$$

where $\mathbf{w}_i^{(l)}$ and $b_i^{(l)}$ are the learnable weight-vector and bias for neuron i in layer l, and f is a (possibly non-linear) activation function [19]. In supervised learning, the NN is trained by comparing predictions to ground-truth labels via a loss function (e.g. cross-entropy), and parameters are optimised by back-propagation and gradient descent.

Graph neural networks (GNNs) generalise NNs to consume graph-structured data $G = (V, E)$, by jointly leveraging a node-feature matrix $\mathbf{X} \in \mathbb{R}^{|V| \times d}$ (for d features) and an adjacency matrix $\mathbf{A}$. Through repeated neighbourhood-aggregation steps—each propagating (also known as *message-passing*) and combining information from a node's neighbours—a GNN learns *embeddings* that encode both local topology and node attributes [20]. These embeddings can serve tasks such as node classification, where a final projection and sigmoid, with a threshold (possibly) tuned for class imbalance, yield per-node labels. Popular aggregation schemes include Graph Convolutional Networks (GCNs) [21] and Graph Attention Networks (GATs) [22].

GCNs perform matrix-based updates across all nodes simultaneously and treat every neighbour equally. GCNs update node representations by aggregating feature information from each node and its neighbours. At each layer l, the representation matrix $\mathbf{H}^{(l+1)}$ is computed from the previous layer $\mathbf{H}^{(l)}$ and the graph structure as:

$$\mathbf{H}^{(l+1)} = \sigma\left(\mathbf{\Delta}^{-1/2}\tilde{\mathbf{A}}\mathbf{\Delta}^{-1/2}\mathbf{H}^{(l)}\mathbf{W}^{(l)}\right),$$

where $\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}$ is the adjacency matrix of the graph with added self-loops (so each node includes its own features in the aggregation), $\mathbf{\Delta}$ is the diagonal degree matrix of $\tilde{\mathbf{A}}$, $\mathbf{W}^{(l)}$ is a learnable weight matrix for layer l, and σ is a (non-linear) activation function. The inclusion of self-loops ensures that nodes can retain and refine their own features across layers, rather than relying solely on neighbouring information. The input to the first layer, $\mathbf{H}^{(0)}$, is typically the initial node feature matrix $\mathbf{X}$.

GATs compute learned attention coefficients [23] to weight each neighbour's contribution differently. Unlike GCNs, which treat all neighbouring nodes equally during aggregation, GATs introduce learnable attention mechanisms to assign different weights to different neighbours. Specifically, for a node i, attention coefficients α_{ij} are computed between i and each of its neighbours j using shared attention:

$$\alpha_{ij} = \frac{\exp\left(\text{LeakyReLU}\left(\mathbf{a}^\top[\mathbf{W}\mathbf{h}_i \parallel \mathbf{W}\mathbf{h}_j]\right)\right)}{\sum_{k\in\mathcal{N}(i)}\exp\left(\text{LeakyReLU}\left(\mathbf{a}^\top[\mathbf{W}\mathbf{h}_i \parallel \mathbf{W}\mathbf{h}_k]\right)\right)},$$

where $\mathbf{W}$ is a learnable weight matrix, $\mathbf{a}$ is a learnable attention vector, $[\cdot \parallel \cdot]$ denotes vector concatenation, and $\mathcal{N}(i)$ is the set of neighbours of node i. LeakyReLU is a non-linear activation function that allows a small negative slope (typically $\epsilon = 0.2$) for negative inputs, helping to avoid dead neurons and improve gradient flow. The updated representation of node i is then computed as a weighted sum of the transformed features of its neighbours: $\mathbf{h}'_i = \sigma\left(\sum_{j\in\mathcal{N}(i)}\alpha_{ij}\mathbf{W}\mathbf{h}_j\right)$. To stabilise the learning process, GATv2 [13] employs multi-head attention: multiple independent attention mechanisms are applied in parallel, and their outputs are concatenated (in intermediate layers) or averaged (in the output layer).

When modelling heterogeneous graphs—where edges carry distinct semantics—one typically employs relation-specific transformations, e.g. via a Heterogeneous Graph Convolution (HGC) module [24], and adds residual connections and normalisation between layers to stabilise training. To prevent overfitting, dropout [25] can be employed after each layer, randomly setting to zero a proportion of the weights, and early stopping to terminate training when the validation loss does not improve for a number of training steps greater than a patience parameter.

4. Predicting Assumptions Acceptance with GNNs

Here we detail our proposed GNN architecture to predict credulous acceptance of assumptions in an ABA framework. The core and novel component is the *dependency graph*, a faithful graph-based representation of an ABAF, that guides the GNN learning of the ABA relationshisps (Section 4.1). The neural machinery adopted to learn to classify the assumption nodes of the dependency graph as *credulously accepted* (IN) or *rejected* (OUT) w.r.t. stable semantics is then detailed in Section 4.2.

4.1. Dependency Graph

Our proposed graph representation contains a node for each atom (assumptions and non-assumptions) and a node for each rule. Also, it contains three distinct edge types. *Support* and *derive edges* connect atom nodes to rule nodes; support edges are labelled $+$, they connect body elements with the respective rule node; derive edges are labelled $\triangleright$; they connect rule node to their respective head elements. *Attack edges* are labelled $-$, they connect contraries and their assumptions.

Definition 4.1. *Let $\mathcal{D} = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^-)$ be an ABAF. The* dependency graph $G_D = (V, E, l)$ *of D is an edge-labelled graph with nodes $V = \mathcal{L} \cup \mathcal{R}$, edges*

$$E = \{(p, r) \mid r \in \mathcal{R}, p \in \mathit{body}(r)\} \cup \{(r, p) \mid r \in \mathcal{R}, p \in \mathit{head}(r)\} \cup \{(p, a) \mid a \in \mathcal{A}, \overline{a} = p\},$$

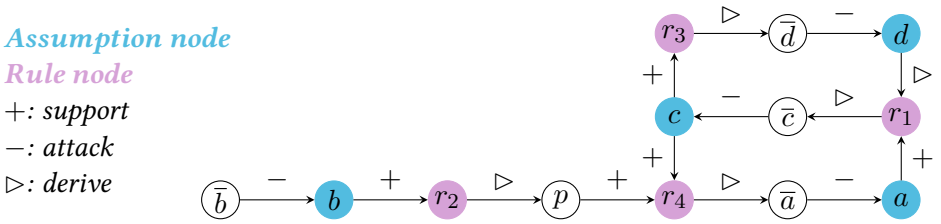
and edge labellings, for $e \in E$,

$$l(e) = \begin{cases} +, & \text{if } e = (p, r), r \in \mathcal{R}, p \in \mathit{body}(r) \\ \triangleright, & \text{if } e = (r, p), r \in \mathcal{R}, p \in \mathit{head}(r) \\ -, & \text{if } e = (p, a), a \in \mathcal{A}, \overline{a} = p \end{cases}$$

Observation 4.2. *The dependency graph $G_\mathcal{D}$ is unique for each $\mathcal{D}$, i.e., for every two ABAFs $\mathcal{D} \neq \mathcal{D}'$, it holds that $G_\mathcal{D} \neq G_{\mathcal{D}'}$.*

An ABAF $\mathcal{D}$ can be fully recovered from its dependency graph $G_{\mathcal{D}}$ since it contains all information about its atoms and rules; thus the semantics are preserved, as for the AF instantiation. In contrast to the AF instantiation, the construction of $G_{\mathcal{D}}$ requires a single pass over the atoms and rules of $\mathcal{D}$.

Example 4.3. *The dependency graph of the ABAF $\mathcal{D}$ from Example 3.4 is shown below:*



We can use $G_{\mathcal{D}}$ to check that b and c jointly derive $\overline{a}$, using rules r_2 and r_4.

Remark 4.4. *Rapberger et al. [26] define a dependency graph for ABA in the spirit of dependency graphs for logic programming [27, 28]. However, their dependency graph does not fully preserve the structure of the ABAF, thus, it is not possible to extract the extensions from a given dependency graph using their approach. In brief, the dependency graph from [26] does not include rule nodes; positive edges (p, q) are introduced whenever p is contained in some rule body of a rule with head q. For example, the rule $(\overline{a} \leftarrow p, c)$ from Example 4.3 would yield the same edges as having two rules $(\overline{a} \leftarrow p), (\overline{a} \leftarrow c)$. In contrast, our dependency graph is unique for every ABAF and carries enough information to preserve the semantics.*

4.2. Neural Architecture

Our GNN operates on the dependency graph's adjacency matrix $\mathbf{W} \in \{0,1\}^{|V|\times|V|}$ and an initial node-feature matrix $\mathbf{F} \in \mathbb{R}^{|V|\times 2}$, where each row F_i contains the in- and out-degrees of node i for each node type (assumptions $\mathcal{A}$, non-assumptions (or claims) $\mathcal{C} = \mathcal{L} \setminus \mathcal{A}$, and rule nodes $\mathcal{R}$). Each node contains a self-loop of type '+' to propagate its features during learning. This is part of the GNN architecture (see Sec. 3.3) and does not interfere with the semantics; note that $G_{\mathcal{D}}$ does not take edges $(s, t) \in \mathcal{L}^2$ into account. In addition, we maintain a learnable embedding matrix $\mathbf{L} \in \mathbb{R}^{|V|\times d_e}$, whose entries are optimised jointly with the rest of the GNN. We show a diagram of the architecture in Fig. 1.

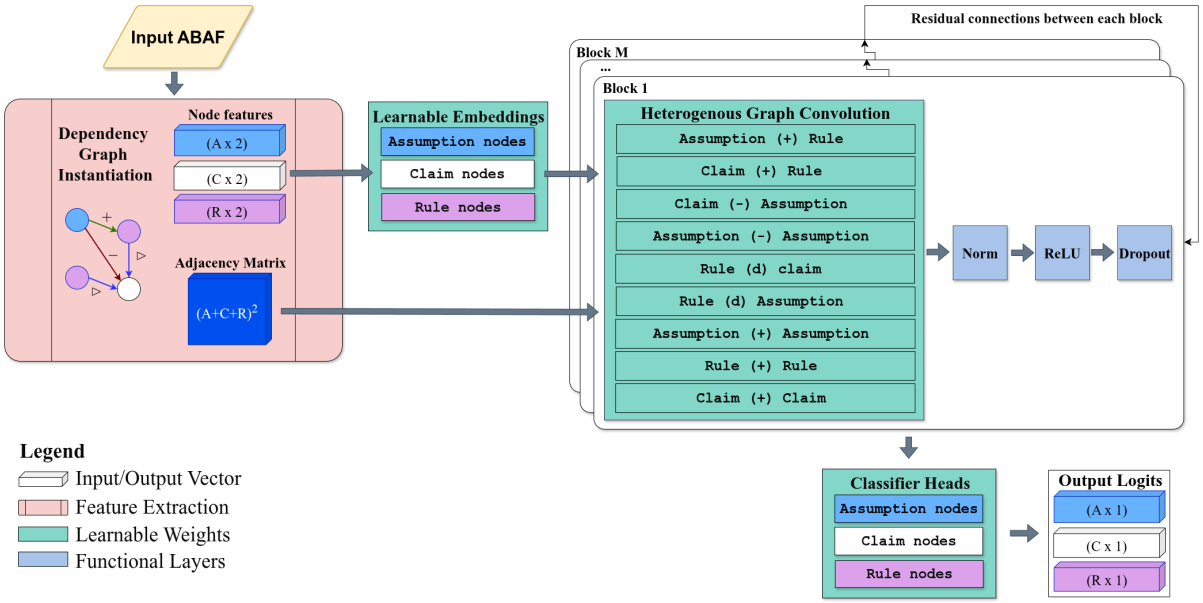


Figure 1: Diagram of the model architecture, including both the feature extraction (using the dependency graph described in Section 4.1) and the GNN described in Section 4.2. The Heterogeneous Graph Convolution (HGC) module would apply Convolutional Layers for ABAGCN, or Attention Layers for ABAGAT.

The core backbone of the GNN consists of M blocks. In block m:

- A HGC layer performs relation-specific neighbourhood aggregation on the current embeddings $H^{(m-1)}$, using either GCN or GAT convolutional kernels (see Section 3 for details).
- The aggregated output is passed through functional layers for stability and regularisation: we use layer-normalisation, a ReLU activation, and dropout with rate δ.
- A residual connection aggregates the block input $H^{(m-1)}$ to its output, yielding $H^{(m)}$.

This design allows distinct transformations per edge type while preserving gradient flow in deep stacks. After M blocks, we extract the rows of $H^{(M)}$ corresponding to assumption, claims and rule nodes and apply a separate linear classifier heads to produce logits $z \in \mathbb{R}^{A+C+R}$, with $A = |\mathcal{A}|, C = |\mathcal{C}|, R = |\mathcal{R}|$.

At inference time, a sigmoid activation followed by a threshold τ (tuned on validation data) yields the predicted probability of credulous acceptance for each assumption.

We train by minimising a weighted binary cross-entropy loss on the assumption logits, using the Adam optimiser with learning rate λ. Class weights compensate for the imbalance between accepted and rejected assumptions (see Table 2). All hyperparameters—including the number of layer blocks M, the embedding dimension d_e, hidden dimension of each convolution layer, the dropout rate δ, the learning rate λ, the batch size, class-weight multiplier, and classification threshold τ—are chosen via Bayesian optimisation [29] with 3-fold cross-validation, as implemented in [30].

Results of the optimisation are provided in Table 1, where we note that the GAT layer achieves a slightly higher validation F1 score (see Section 5 for more details on data and evaluation). Interesting observations from Table 1 are that the optimal dropout rate, batch sizes and early stopping thresholds reflect the role of attention in reducing the number of parameters (higher dropout) and the number of examples needed (lower batch) but higher tendency to over-fitting (higher patience needed). We also note that the number of layers was strongly positively correlated with better performance, indicating that more than 10 layers would probably improve performance. We leave this as future work because of time and compute constraints.

Table 1
Tested hyperparameter ranges and best configurations for GCN and GAT

Parameter	Values tested	GCN (best)	GAT (best)
Validation F1 Score	—	0.6682	0.6709
Embedding dimension	16, 32, 64, 128, 256	32	64
Hidden dimension	32, 64, 128, 256, 512	32	64
Number of layers blocks	2–10	10	10
Dropout rate	0–0.5 (uniform)	0.0294	0.2198
Learning rate	10^{-2}–10^{-4} (uniform)	0.0086	0.0066
Batch size	16, 32, 64, 128	128	64
Class imbalance weight	1.25, 1.5, 1.75, 2.0, 2.25	2.25	1.75
Classification threshold	0.20–0.85 (step 0.05)	0.50	0.45
Early stopping patience	10, 30, 50, 70, 90	10	30

5. Empirical Evaluation

In this section we assess the effectiveness of our proposed GNN models—both the convolutional (ABAGCN) and attention-based (ABAGAT) variants—against our AFGCNv2-based baseline on two tasks over ABAFs under stable semantics:

1. **Credulous acceptance classification**: predicting, for each assumption, whether it belongs to any stable extension.
2. **Extension reconstruction**: recovering the full set of accepted assumptions (i.e. a predicted extension) and comparing it to the ground-truth stable extensions.

Table 2
Parameter ranges for ICCMA benchmarks and additional synthetic data

Parameter	ICCMA	Generated
Acceptance Rate (with at least one extension)	36.7% (60.3%)	32.5% (53.4%)
Number of atoms	25, 100, 500, 2000, 5000	25, 50, 75, 100, 250, 500, 750, 1000, 2000, 3000, 4000, 5000
Assumption proportion	10%, 30%	25%, 40%, 50%, 60%, 75%
Max. rules per derived atom	5, 10	2, 4, 8, 16
Max. rule-body length	5, 10	2, 4, 8, 16
Cycle cap	—	0.01, 0.03, 0.05, 0.07, 0.09, 1.0
Total instances (post-filtering)	400 (380)	48 000 (19 500)

For the first task we report node-level precision, recall, F1 and accuracy. For the second task we treat each stable extension as a set and measure how well our predicted sets match ground truth using extension-level F1 (equivalently, the F1 of the predicted versus true assumption sets).

5.1. Data

ICCMA 2023 included an ABA track–but not an approximate ABA one–whose benchmark comprised of 400 ABAFs defined by four parameters: number of atoms, assumption proportion, maximum rules per atom, and maximum rule-body length. Ground-truth labels are obtained by running ASPForABA [31][2] on every test instance with timeout threshold 10 min, yielding 380 benchmark ABAFs with exact annotations within the ICCMA data.

To obtain a larger training corpus, we used the ABAF generator from [31] to produce 48,000 flat ABAFs (47,794 after timeout filtering when producing labels), extending the parameter ranges. Parameters are shown in Table 2. To match the original assumptions acceptance rate (36.7% overall, 60.3% within ABAFs with at least one stable extension), we curated a subset of 19,500 ABAFs (13,000 with at least one accepted assumption, 6,500 with none), yielding 32.5% overall acceptance and 53.4% within ABAFs with at least a solution.

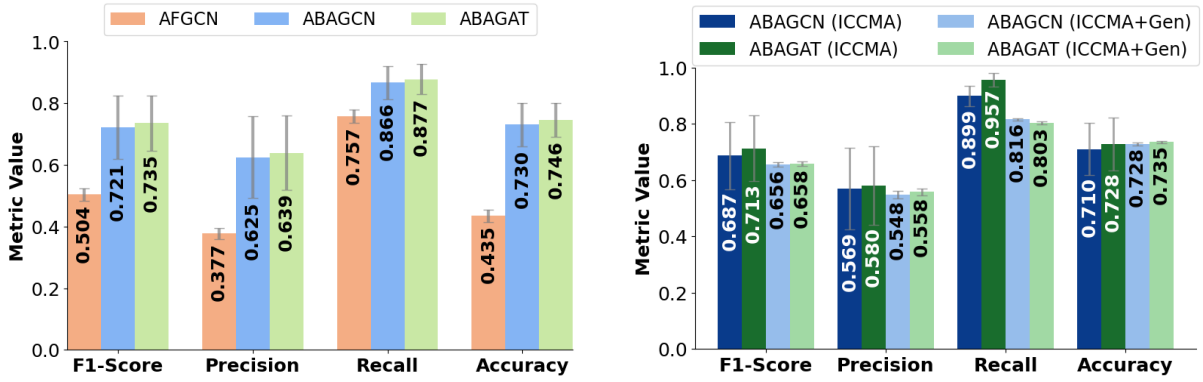
Finally, we applied a stratified 75:25 train–test stratified over three groups—small ICCMA (25–100 atoms), full ICCMA, and our generated data. The stratification followed from seeking a fair comparison against AFGCNv2, which cannot handle ABAFs with more than 100 atoms. All results are therefore shown on the unseen test sets from these groups: ICCMA small, ICCMA (full) and ICCMA+Generated.

5.2. Baseline

Our baseline implements a three-stage pipeline that reduces ABAF credulous acceptance under stable semantics to an AF classification task:

1. **ABAF-AF translation:** Given an input ABAF, we use the method from [18] and implemented in the AcBar toolkit [32][3] to convert the input ABAF into a (poly-sized) AF in polynomial time.
2. **Argument classification via AFGCNv2:** The generated AF is represented by its adjacency matrix and graph-level features, then fed into the AFGCNv2 solver (see Section 2) using pre-trained weights for credulous acceptance under stable semantics. To ensure a fair comparison with other models, we disable its grounded-extension pre-check and apply a classification threshold—tuned on a held-out validation set—to the sigmoid output so that any argument with score above the threshold is marked as accepted.
3. **Element acceptability recovery:** Finally, we derive assumption acceptance in the original ABAF: an assumption a is declared accepted if the argument $\{a\} \vdash a$ is accepted.

[2]Available at https://bitbucket.org/coreo-group/aspforaba.
[3]Available at https://bitbucket.org/lehtonen/acbar



(a) AF vs ABA GNNs on ICCMA data with $|\mathcal{L}| < 100$

(b) CGN vs GAT on ICCMA and generated data

Figure 2: Model Comparison according to F1, Precision, Recall and Accuracy on different cuts of data: small ICCMA ABAFs (with less than 100 elements) in panel (a) to be able to compare to the AFGCN baseline; Comparison between all ICCMA ABAFs and the test set from our augmented training data in panel (b).

This baseline thus leverages existing AF-based GNN techniques for assumption-based frameworks, providing a direct point of comparison for ur native ABAF solver.

5.3. Credulous Acceptance Classification

We first evaluate how well each model predicts assumption-level acceptance under stable semantics. Figure 2a shows that both dependency-graph GNNs substantially outperform the AF baseline on the small ICCMA dataset ($|\mathcal{L}| < 100$). F1 increases from 0.5 (AFGCN) to 0.72 (ABAGCN) and 0.74 (ABAGAT); recall climbs by about 10 points; precision improves by about 25 points; and accuracy rises by over 30 points. The attention-based ABAGAT achieves the highest results across all metrics on this test set.

In Figure 2b we examine the effect of augmenting the ICCMA training data with our generated instances. Compared to the full ICCMA test set, both ABAGCN and ABAGAT see modest gains in accuracy but incur a notable drop in precision and recall, resulting in a decrease in F1. This pattern indicates that synthetic augmentation possibly produced more challenging ABAFs than the ones in the competition. Crucially, despite this worsening in performance, both GNN variants maintain strong performance across the held-out ICCMA and generated test split, demonstrating robust generalisation to larger and more diverse ABA frameworks.

Comparing the small ICCMA results in Figure 2a to the (full) ICCMA bars in Figure 2b, F1 dips by under one percentage point, precision by roughly two points and accuracy by about one point on the full dataset, while recall actually climbs by two to six points, for ABAGCN and ABAGAT, respectively. In other words, on larger ABAFs both GNNs trade a bit of overall and positive-class fidelity for stronger coverage of accepted assumptions.

5.4. Extension Reconstruction

For our second experiment, we test a poly-time algorithm that uses our approximate ABA models to calculate a stable extension. The algorithm works as follows, see the sketch in Algorithm 1. First, given an ABAF $\mathcal{D}$, we use the credulous acceptance prediction obtained from, e.g., ABAGAT to pick an assumption a^* (line 3). Intuitively, this pick means that a^* is *accepted* in $\mathcal{D}$. To make our next guess, we modify the ABAF $\mathcal{D}$ accordingly so that the extensions of the modified ABAF $\mathcal{D}'$ (under projection) correspond to the extensions of $\mathcal{D}$ that contain a^* (line 4). Since our prediction is imperfect, this modification may have introduced some inconsistencies in $\mathcal{D}'$, which we check in line 5. If the check passes, we add a^* to our extension $\mathcal{E}$. We outline $\textsc{modify}(\mathcal{D}, a^*)$ in more detail. When removing from the ABAF an assumption a^* predicted IN, we want to ensure that the modified ABAF $\mathcal{D}'$ satisfies

$$\{S \cap \mathcal{A} \mid S \in stb(\mathcal{D}')\} = \{S \setminus \{a^*\} \mid S \in stb(\mathcal{D}),\ a^* \in S\}, \tag{1}$$

Require: ABAF $\mathcal{D} = (\mathcal{L}, \mathcal{R}, \mathcal{A}, {}^-)$, predictor $\mathcal{M}$

1: Initialize extension $\mathcal{E} \leftarrow \emptyset$
2: **while** $\mathcal{A} \neq \emptyset$ **do**
3: $\quad a^* \leftarrow \arg\max_{a \in \mathcal{A}} P_{\mathcal{M}}(a)$
4: $\quad \mathcal{D}' \leftarrow \textsc{modify}(\mathcal{D}, a^*)$
5: $\quad$ **if** $\textsc{isConflicting}(\mathcal{D}')$ **then**
6: $\quad\quad$ **break**
7: $\quad$ **else**
8: $\quad\quad \mathcal{E} \leftarrow \mathcal{E} \cup \{a^*\}$
9: **return** $\mathcal{E}$

Algorithm 1: Extension reconstruction sketch

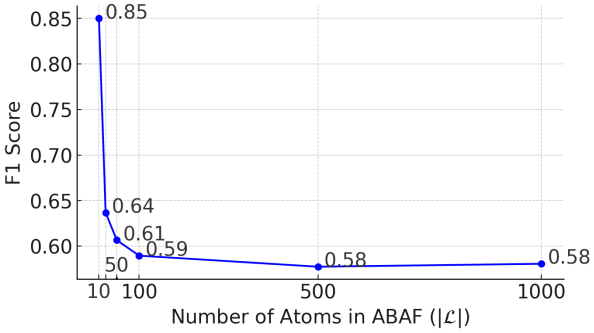


Figure 3: F1 score of the extension by ABAF size.

that is, the stable extensions of $\mathcal{D}'$ projected on $\mathcal{A}$ correspond to the extensions of $\mathcal{D}$ that contain a^*. We project onto $\mathcal{A}$ because the modification requires the introduction of some dummy-assumptions to ensure that rules that derive the contrary of a^* act as constraints. Concretely, given an ABAF $\mathcal{D}$ and an assumption a^*, $\textsc{modify}(\mathcal{D}, a^*)$ performs the following steps:

1. We remove the assumption a^* and all occurrences of it; that is, we let $\mathcal{A}' = \mathcal{A} \setminus \{a^*\}$ and we replace any rule r with $r' = head(r) \leftarrow body(r) \setminus \{a^*\}$.
2. We modify each rule r with $head(r) = \overline{a^*}$: for each such rule r, we introduce a new dummy assumption d_r and replace r with $r' = \overline{d_r} \leftarrow body(r) \cup d_r$. This modification ensures that not all body elements of r can be true (accepted or derived) at the same time.[4]
3. If a^* is the contrary of another assumption c, then c is rejected. We remove c and any rule r with $c \in body(r)$ from the ABAF.
4. Lastly, we remove all facts from $\mathcal{D}$: for each rule r with $body(r) = \emptyset$, we remove r from the rule set and we remove $p = head(r)$ from the body of each rule r', i.e., we replace r' with $r'' = head(r') \leftarrow body(r') \setminus \{p\}$. If p is the contrary of an assumption c, the assumption is attacked and we proceed as in Step 3. We repeat this process until there are no remaining facts.

It can be checked that if a^* is credulously accepted with respect to stable semantics, then the algorithm $\textsc{modify}(\mathcal{D}, a^*)$ satisfies the statement in Equation 1. Otherwise, in case a^* is not credulously accepted in $\mathcal{D}$, Algorithm 1 will encounter a rule of the form $\overline{d_r} \leftarrow d_r$ which indicates an inconsistency in $\mathcal{D}$. We perform this inconsistency check in $\textsc{isConflicting}(\mathcal{D}')$ by checking the rules of the ABAF $\mathcal{D}$ (line 5 of Algorithm 1) and return the partially computed set as soon as we encounter an inconsistency.

Note that $\textsc{modify}(\mathcal{D}, a^*)$ runs in polynomial time; naively, the computation requires a loop over all rules in $\mathcal{R}$, for each $p \in \mathcal{L}$; thus, Algorithm 1 is in P.

Figure 3 illustrates that our constructive extension reconstruction method achieves very high F1 (0.85) on small ABAFs (10 atoms), but performance drops sharply to around 0.60 by 50–100 atoms and then plateaus at approximately 0.58 for larger frameworks. This decline reflects how early misclassifications by the GNN propagate through subsequent modification steps, reducing both precision and recall. Nonetheless, sustaining an F1 above 0.5 even on ABAFs with 1,000 atoms indicates that the approach remains moderately robust at scale, highlighting opportunities for future work on error-correction or lookahead strategies to mitigate this degradation.

We also compare runtimes on the 15 most challenging ABAFs (4,000–5,000 atoms) from our generated dataset, held out in the test split. The exact ASPForABA solver averaged 435s per instance, whereas our approximate extension reconstruction ran in 192s on average—2.3× faster—while still achieving F1 of 0.68 and accuracy of 0.77 on those large frameworks. These results demonstrate that GNN-driven approximate reasoning can yield substantial speed-ups over exact methods, maintaining a similar level predictive performance as compared to the easier single assumption prediction task.

[4]This modification resembles a constraint, i.e., a rule with empty head in logic programming, cf. [33].

6. Conclusion

We have demonstrated that heterogeneous GNNs over our dependency-graph encoding can accurately and efficiently approximate credulous acceptance in ABA under stable semantics. Both ABAGCN and ABAGAT outperform an AF-based GNN baseline—achieving node-level F1 of 0.65 overall and up to 0.71 on small frameworks—while our poly-time extension-reconstruction procedure reconstructs stable sets with F1 greater than 0.85 on small ABAFs and maintains F1 of about 0.58 at scales of 1,000 atoms. Crucially, on the 4,000–5,000-atom ABAFs where exact ASPForABA averaging 435s, our approximate enumeration runs in 192s (2.3× faster) with F1 of 0.68, underscoring substantial runtime gains without sacrificing predictive quality. These findings show that neural approximation can bridge the gap between accuracy and tractability in structured argumentation.

Future work includes integrating lightweight symbolic checks to boost precision (e.g. grounded semantics as done in [12]), extending the approach to other semantics, including admissible, complete, and preferred, where GNNs may be able to exploit local structural patterns even more efficiently. Additionally we aim at lifting the flatness restriction to handle fully general ABAFs and cover applications of ABA where non-flat models are used [4]. Together, these directions promise more scalable and interpretable reasoning tools for complex argumentative scenarios.

Acknowledgments

Rapberger and Russo were funded by the ERC under the ERC-POC programme (grant number 101189053) while Toni under the EU's Horizon 2020 research and innovation programme (grant number 101020934); Toni also by J.P. Morgan and by the Royal Academy of Engineering under the Research Chairs and Senior Research Fellowships scheme.

References

[1] P. M. Dung, On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games, Artif. Intell. 77 (1995) 321–357.

[2] L. Amgoud, H. Prade, Using arguments for making and explaining decisions, Artif. Intell. 173 (2009) 413–436. doi:10.1016/j.artint.2008.11.006.

[3] K. Cyras, A. Rago, E. Albini, P. Baroni, F. Toni, Argumentative XAI: A survey, in: Proc. of IJCAI, 2021, pp. 4392–4399. doi:10.24963/IJCAI.2021/600.

[4] F. Russo, A. Rapberger, F. Toni, Argumentative causal discovery, in: Proc. of KR, 2024, pp. 938–949. doi:10.24963/KR.2024/88.

[5] K. Čyras, Q. Heinrich, F. Toni, Computational complexity of flat and generic assumption-based argumentation, with and without probabilities, Artif. Intell. 293 (2021) 103449. doi:https://doi.org/10.1016/j.artint.2020.103449.

[6] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, G. Monfardini, The graph neural network model, IEEE Transactions on Neural Networks 20 (2009) 61–80. doi:10.1109/TNN.2008.2005605.

[7] I. Kuhlmann, M. Thimm, Using graph convolutional networks for approximate reasoning with abstract argumentation frameworks: A feasibility study, in: Proc. of SUM, 2019, pp. 24–37. doi:10.1007/978-3-030-35514-2_3.

[8] P. Cibier, J.G. Mailly, Graph convolutional networks and graph attention networks for approximating arguments acceptability, in: Proc. of COMMA, 2024, pp. 25–36. doi:10.3233/FAIA240307.

[9] L. Malmqvist, T. Yuan, P. Nightingale, S. Manandhar, Determining the acceptability of abstract arguments with graph convolutional networks, in: Proc. of COMMA, 2020, pp. 47–56. URL: https://ceur-ws.org/Vol-2672/paper_5.pdf.

[10] J.M. Lagniez, E. Lonca, J.G. Mailly, CoQuiAAS: A constraint-based quick abstract argumentation solver, in: Proc. of ICTAI, 2015, pp. 928–935. doi:10.1109/ICTAI.2015.134.

[11] B. Perozzi, R. Al-Rfou, S. Skiena, Deepwalk: online learning of social representations, in: Proc. of KDD, 2014, p. 701–710. doi:10.1145/2623330.2623732.

[12] L. Malmqvist, T. Yuan, P. Nightingale, Approximating problems in abstract argumentation with graph convolutional networks, Artif. Intell. 336 (2024) 104209. doi:10.1016/J.ARTINT.2024.104209.
[13] S. Brody, U. Alon, E. Yahav, How attentive are graph attention networks?, in: Proc. of ICLR, 2022. URL: https://openreview.net/forum?id=F72ximsx7C1.
[14] D. Craandijk, F. Bex, AGNN: A deep learning architecture for abstract argumentation semantics, in: Proc. of COMMA, 2020, pp. 457–458. doi:10.3233/FAIA200532.
[15] A. Bondarenko, P. M. Dung, R. A. Kowalski, F. Toni, An abstract, argumentation-theoretic approach to default reasoning, Artif. Intell. 93 (1997) 63–101. doi:10.1016/S0004-3702(97)00015-5.
[16] K. Cyras, X. Fan, C. Schulz, F. Toni, Assumption-Based Argumentation: Disputes, Explanations, Preferences. In Handbook of Formal Argumentation., College Publications, 2018.
[17] P. M. Dung, P. Mancarella, F. Toni, Computing ideal sceptical argumentation, Artif. Intell. 171 (2007) 642–674. doi:10.1016/J.ARTINT.2007.05.003.
[18] T. Lehtonen, A. Rapberger, M. Ulbricht, J. P. Wallner, Argumentation frameworks induced by assumption-based argumentation: Relating size and complexity, in: Proc. of KR, 2023, pp. 440–450. doi:10.24963/KR.2023/43.
[19] C. M. Bishop, Pattern recognition and machine learning, 5th Edition, Information science and statistics, Springer, 2007. URL: https://www.worldcat.org/oclc/71008143.
[20] M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, P. Vandergheynst, Geometric deep learning: Going beyond euclidean data, IEEE Signal Process. Mag. 34 (2017) 18–42. doi:10.1109/MSP.2017.2693418.
[21] T. N. Kipf, M. Welling, Semi-supervised classification with graph convolutional networks, in: Proc. of ICLR, 2017. URL: https://openreview.net/forum?id=SJU4ayYgl.
[22] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Liò, Y. Bengio, Graph attention networks, in: Proc. of ICLR, 2018. URL: https://openreview.net/forum?id=rJXMpikCZ.
[23] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, I. Polosukhin, Attention is all you need, in: Proc. of NeurIPS, 2017, pp. 5998–6008. URL: https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html.
[24] X. Wang, H. Ji, C. Shi, B. Wang, Y. Ye, P. Cui, P. S. Yu, Heterogeneous graph attention network, in: Proc. of WWW, 2019, pp. 2022–2032. doi:10.1145/3308558.3313562.
[25] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: a simple way to prevent neural networks from overfitting, J. Mach. Learn. Res. 15 (2014) 1929–1958. doi:10.5555/2627435.2670313.
[26] A. Rapberger, M. Ulbricht, J. P. Wallner, Argumentation frameworks induced by assumption-based argumentation: Relating size and complexity, in: Proc. of NMR, 2022, pp. 92–103. URL: https://ceur-ws.org/Vol-3197/paper9.pdf.
[27] K. Konczak, T. Linke, T. Schaub, Graphs and colorings for answer set programming, Theory Pract. Log. Program. 6 (2006) 61–106. doi:10.1017/S1471068405002528.
[28] J. Fandinno, V. Lifschitz, Positive dependency graphs revisited, Theory Pract. Log. Program. 23 (2023) 1128–1137. doi:10.1017/S1471068422000333.
[29] J. Snoek, H. Larochelle, R. P. Adams, Practical bayesian optimization of machine learning algorithms, in: Proc. of NeurIPS, 2012, pp. 2960–2968. URL: https://proceedings.neurips.cc/paper/2012/hash/05311655a15b75fab86956663e1819cd-Abstract.html.
[30] L. Biewald, Experiment tracking with weights and biases, 2020. URL: https://www.wandb.com/.
[31] T. Lehtonen, A. Rapberger, F. Toni, M. Ulbricht, J. P. Wallner, Instantiations and computational aspects of non-flat assumption-based argumentation, in: Proc. of IJCAI, 2024, pp. 3457–3465. doi:10.24963/ijcai.2024/383.
[32] T. Lehtonen, A. Rapberger, M. Ulbricht, J. P. Wallner, Acbar–atomic-based argumentation solver, ICCMA 2023 42 (2021) 16.
[33] A. Rapberger, M. Ulbricht, F. Toni, On the correspondence of non-flat assumption-based argumentation and logic programming with negation as failure in the head, in: Proc. of NMR, 2024, pp. 112–121. URL: https://ceur-ws.org/Vol-3835/paper12.pdf.

Argumentation-based Data-Leakage Analysis

Bettina Fazzinga[1,*,†], Sergio Flesca[2,†], Filippo Furfaro[2,†] and Giuseppina Monterosso[2,†]

[1]DiCES - Università della Calabria
[2]DIMES - Università della Calabria

Abstract

We propose a novel framework for supporting the analysis of data leakage events based on abstract argumentation. The goal is to provide an analyst with a formal tool to reason about which agents might be responsible for the unauthorized disclosure of sensitive information. The reasoning process focuses on identifying coalitions of agents who, by pooling their possible knowledge, can justify the leaked arguments. This model captures key characteristics of real-world scenarios such as insider trading and investigative journalism, where the act of disclosure is typically backed by a coherent rationale.

Keywords

Argumentation, Data Leakage, Uncertain reasoning

1. Introduction

In his seminal work, Dung introduced *Abstract Argumentation Frameworks (AAF)* [1] as a general formalism for representing and evaluating arguments and their interactions. An AAF consists of a set of abstract arguments and a binary attack relation, and allows one to determine which arguments—or sets of arguments—can be considered acceptable under different semantics. Due to their flexibility and expressive power, AAFs have been successfully applied not only in core argumentation theory, but also in areas such as process mining [2], medical applications [3], decision support systems [4], and E-democracy [5]. Many approaches have been recently proposed for extending AAFs to deal with uncertainty. In particular, in the *incomplete Abstract Argumentation Frameworks (iAAFs)* [6], every argument/attack is labeled as certain or uncertain, where everything labeled as "certain" must be considered as present in the reasoning, while, for each argument or attack labeled as "uncertain", the reasoner must consider that it may be present or absent.

In this paper, we explore the application of iAAFs in the domain of information security, specifically for analyzing data leakage events: cases where a collection of sensitive arguments has been disclosed without authorization. Our aim is to support an analyst in reasoning about who might be responsible for the leakage, based on partial and uncertain domain knowledge. The analyst's domain is composed of a set of arguments, including those that were leaked. The suspects are all agents involved in the domain, and the analyst has uncertain information about which arguments each agent might know. This uncertainty arises from the agents' roles, education, workplace, and social or familial relationships.

A key assumption in our approach is that the leaked arguments are not arbitrary; rather, they are expected to be justifiable by the person or group who disclosed them. This reflects practices in domains such as insider trading, where leaked information must be accurate and actionable to be of value, and investigative journalism, where published arguments must withstand potential legal scrutiny.

Given this assumption, the analyst's reasoning reduces to identifying coalitions of agents whose combined (possible) knowledge suffices to justify the leaked arguments within an abstract argumentation framework, as shown in the example below.

Example 1. *Suppose that an analyst is investigating the leakage of a sensitive piece of information suggesting that the stock value of the company* Omega *is expected to rise. The analyst suspects three agents: A (an internal accountant with access to some internal documents), B (an assistant to the CEO with access to some internal communications), and C (an internal employee part of the HR department). According to the analyst, the agents are aware of the following arguments:*

Arg&App 2025: International Workshop on Argumentation and Applications, November 2025, Melbourne, Australia
*Corresponding author.
†These authors contributed equally.
bettina.fazzinga@unical.it (B. Fazzinga); sergio.flesca@unical.it (S. Flesca); filippo.furfaro@unical.it (F. Furfaro); giusy.monterosso@dimes.unical.it (G. Monterosso)

- ***Agent A*** *- Argument x_1: "Since there will be an increase in profits of Omega for the current quarter and the outlook for maintaining this increase is positive, Omega's stocks will rise in value in the short term."*
- ***Agent A*** *- Argument x_2: "Since, as reported in the internal document $\#1324$ of the 1st February 2025, Omega is experiencing serious issues with a strategic client, it is likely that future profits will drop sharply."*
- ***Agent B*** *- Argument x_3: "Since, as reported in the recent CEO's note $\#2221$ of the 15th February 2025, the issues with the strategic client have been resolved, a new agreement has been reached and the contract has been successfully renewed."*
- ***Agent C*** *- Argument x_4: "Since Omega recently hired a new CFO with a strong background in cost optimization, the operational efficiency may improve in the long term."*

In this example, the leaked data corresponds to the conclusion of argument x_1. Therefore, the analyst focuses on investigating whether x_1 can be justified by the knowledge of one or more of the suspected agents. In this case, x_1 cannot be justified by the knowledge of a single agent. In fact, since x_2 contradicts x_1, agent A alone cannot be the source of the leak, as x_1 would not be justified from their individual perspective. Agents B and C cannot be responsible either, as they are not aware of x_1. On the other hand, if we conjecture that A and B collaborated and combined their knowledge, the situation changes: B's argument x_3 attacks A's argument x_2, which attacks x_1, thus x_1 is accepted in the joint argumentation framework. Hence, the coalition $\{A, B\}$ can be reasonably considered as a strong suspect for the leak. Now, consider the larger coalition $\{A, B, C\}$, that has access to arguments x_1, x_2, x_3, and x_4. Since x_4 does not attack or defend any other argument (as it deals with long-term predictions independent from the other arguments), the justification of x_1 remains unchanged. Thus, the knowledge of the coalition $\{A, B, C\}$ can still justify x_1, but the presence of the argument known to agent C is not necessary. Therefore, there is no strong reason to suspect the whole coalition $\{A, B, C\}$ as a responsible in the leakage. This example illustrates the importance of focusing on minimal coalitions ($\{A, B\}$, in this case) that are sufficient to justify the leaked argument. Considering larger, non-minimal coalitions may lead to wrongly implicating uninvolved agents.

In order to support the investigative reasoning in the data-leakage scenario of Example 1, we introduce the framework *ARDA* (*ARgumentation-based Data-leakage Analysis*). ARDA is a generalization of iAAFs that enables the subjective views of the various agents suspected as contributors to the leakage to be represented, and allows the analyst to specify which terms of these views are known for certain and which terms are possible but not certain. Over ARDA, we define the fundamental problems at the basis of the investigative reasoning, and we address several properties characterizing the reasoning itself.

Overall, our contribution is a formal and principled basis for supporting the investigation in the data leakage scenario, that offers a novel application of argumentation theory to investigative tasks in information security and intelligence analysis.

2. Preliminaries

We assume that the reader is familiar with Dung's AAFs, in particular with the notion of extension and of credulous and skeptical acceptance. We assume that the reasoning is performed under any semantics in the set {*stable, grounded, complete, preferred*}, and the semantics is implied all through the paper.

We now recall the notion of *incomplete AAF* [6], which extends the notion of AAF with the possibility of specifying that some arguments and/or attacks of the modeled dispute may not occur.

Definition 1 (iAAF). *An* incomplete Abstract Argumentation Framework *is a tuple $\langle A, A^?, D, D^? \rangle$, where A and $A^?$ are disjoint sets of arguments, and D and $D^?$ are disjoint sets of attacks between arguments in $A \cup A^?$. The arguments in A are said to be* certain *(i.e. they are definitely known to exist), while those in $A^?$* uncertain *(i.e. it is not known for sure if they occur in the argumentation or not). The attacks in D are said to be* certain *(i.e. they are definitely known to exist, if both the incident arguments exist), while those in $D^?$* uncertain *(i.e. it is not known for sure if they exist or not, and they can exist only if both the incident arguments exist).*

An iAAF compactly represents the alternative scenarios for the argumentation, called *completions*.

Definition 2 (Completion). *A completion for an iAAF $IF = \langle A, A^?, D, D^? \rangle$ is an AAF $F = \langle A', D' \rangle$ where $A \subseteq A' \subseteq (A \cup A^?)$ and $D \cap (A' \times A') \subseteq D' \subseteq (D \cup D^?) \cap (A' \times A')$.*

In [7], the notion of accepted argument has been adapted to iAAFs by interweaving the *possible* and *necessary* perspectives (which differ in how the presence of multiple **completions** is taken into account) with the *credulous* and *skeptical* perspectives (which differ in how the presence of multiple **extensions** is taken into account), as reported below.

Definition 3 (Possibly and necessarily accepted arguments)**.** *An argument a of an iAAF IF is said to be possibly (resp., necessarily) X-accepted (with $X \in \{$credulously, skeptically$\}$ if it is an accepted argument in at least one (resp., in every) completion of IF.*

3. The ARDA framework

In this section, we introduce the *ARgumentation for Data-leakage Analysis* (*ARDA*) framework. First, we provide details about the scenario in which the data leakage occurred, specifically what information we assume is available to the analyst. Then, we formally define the reasoning framework, in which all this information is appropriately encoded and used to support the identification of the leakers.

The analyst's investigative hypothesis is that a group of agents maliciously formed a coalition to decide whether to disclose certain information. The analyst focuses attention on a set L of agents (called *candidate leakers*) who, being aware of various pieces of information related to the leakage, could have been part of the coalition. In conducting the investigation, the analyst operates under the assumption that the coalition, when deliberating on whether to disseminate certain information, must be convinced of its verifiability (as discussed in the introduction, with reference to insider trading and investigative journalism scenarios). Therefore, the analyst carries out the investigation by modeling the collaboration among the coalition members in argumentative terms. Specifically, the collaboration is modeled as an *argumentative framework* where: (i) each agent contributes by sharing their own perspective, namely proposing a set of arguments and a set of attacks among the arguments that emerge during the collaboration; (ii) the content of the leakage turns out to be an accepted argument.

More specifically, for each candidate leaker, the analyst has some information regarding their knowledge, expertise, and ideological stance; using this information, the analyst can formalize the contributions (i.e., arguments and attacks) that each agent might offer during the malicious collaboration with the other agents. In formal terms, we denote the set of candidate leakers as L, the set of arguments that may be claimed by at least one of the candidate leakers as A, and the set of attacks that may be perceived by at least one candidate leaker as D.

Starting from this, we formalize the profile that the analyst has of each candidate leaker by means of a partial function $P_l : A \times D \to \{$*yes*, *maybe*$\}$, which encodes the possible contributions that, according to the analyst, each candidate leaker l makes to the coalition. Specifically, for a given argument or attack x, $P_l(x) =$ *yes* (resp., *maybe*) means that, in the analyst's view, *it is certain* (resp., *it may happen*) that l will contribute to the coalition by proposing x.

The tuple $\mathcal{AF} = \langle L, A, D, \mathcal{P}, a \rangle$, where $\mathcal{P} = \{P_l | l \in L\}$, summarizing the information on which the analyst will rely as the basis for their investigation will be called "*ARDA framework*".

3.1. Reasoning over the ARDA framework

Given an ARDA framework $\mathcal{AF} = \langle L, A, D, \mathcal{P}, a \rangle$, in order to identify possible coalitions that are responsible for the leakage, it is necessary to formalize the fact that the argumentative reasoning performed over the views (i.e. sets of arguments and attacks) shared by the coalition members leads to the justification of the leaked argument a. In turn, it is necessary to consider both the uncertainty with which the analyst models each view (as encoded in the profile P_l of each candidate leaker l) and the possible differences between the profiles of coalition members. In essence, the outcome of a coalition's reasoning should be based on the set of arguments and attacks that are proposed by some member of the coalition and recognized by others as part of the reasoning process. According to this rationale:

- if an argument/attack x is proposed with certainty by all members of the coalition, then it is considered to be *certainly present*;
- if an argument/attack x is not proposed (neither certainly nor possibly) by any member of the coalition, then it is considered to be *certainly absent*;

- in all other cases, there is at least a chance that some agent may propose x, but whether or not it is considered in the reasoning process depends on whether the coalition agrees on considering it. In such cases, x is treated as an *uncertain argument/attack*. That is, in the analyst's attempt to reconstruct the actual reasoning carried out by the coalition, both the case where x is present and the case where it is absent will be considered.

A suitable and compact way to represent this scenario is to rely on the paradigm of iAAFs. Specifically, following the above considerations, we model the collaboration of a coalition $C \subseteq L$ through the so called "*induced iAAF*" $F_C = \langle A_C, A_C^?, D_C, D_C^? \rangle$, where:

- $A_C = \{a \in A \mid \forall l \in C\ P_l(a) = \text{yes}\}$,
- $A_C^? = \big\{a \in A \setminus A_C \mid \exists l \in C \text{ s.t. } P_l(a) \in \{\text{yes, maybe}\}\big\}$,
- $D_C = \big\{d \in D \cap ((A_C \cup A_C^?) \times (A_C \cup A_C^?)) \mid \forall l \in C\ P_l(d) = \text{yes}\}$,
- $D_C^? = \big\{d \in (D \cap ((A_C \cup A_C^?) \times (A_C \cup A_C^?))) \setminus D_C \mid \exists l \in C \text{ s.t. } P_l(d) \in \{\text{yes, maybe}\}\big\}$.

On this basis, we introduce the concept of *Valid Leakers Coalition* w.r.t. an argument a, that is the fundamental notion for the investigative reasoning performed by the analyst.

Definition 4 ((Minimal) Valid Leakers Coalition (VLC))**.** *Given an argument $a \in A$, a coalition of candidate leakers C is a possible (resp., necessary)* Valid Leakers Coalition *(w.r.t. a) iff a is possibly (resp., necessarily) accepted in $\mathcal{F}_C$ under the credulous perspective. A Valid Leakers Coalition C is said to be* Minimal *(w.r.t. a) iff for each $C' \subset C$ it holds that C' is not a Valid Leakers Coalition (w.r.t. a).*

Thus, a coalition C is a *Valid Leakers Coalition* (w.r.t. a) if the argumentative reasoning carried out over the combination of their individual knowledge leads to a being justified. Therefore, from the analyst's perspective, a VLC is a coalition that can be suspected of having caused the leakage of a. The more refined notion of MVLC aims to isolate the minimal coalitions of agents on which to focus suspicion, in accordance with the general principle of not incriminating suspects (at least at first glance) unless their contribution to the coalition is mandatory (i.e. the rest of the coalition would not be able to accomplish the criminal act by themselves).

The *necessary* and *possible* variants of VLC arise from the uncertainty regarding which arguments and attacks are actually considered by the coalition. Specifically, in the presence of this uncertainty, the *necessary* and the *possible* variants of VLC encode a more and a less "guarantor" approach to decide whether a coalition can be considered capable of having leaked the argument in question, respectively.

Observe that VLCs and MVLCs are defined under the credulous perspective of the acceptance: this is because we believe that credulous acceptance is strong enough to support a well-founded suspicion. However, all the theoretical results characterizing the reasoning under the credulous perspective reported in what follows would still hold if the skeptical perspective were adopted.

On the basis of the notions of VLC and MVLC, we now introduce some fundamental problems supporting the investigative reasoning over the ARDA framework. We start with the *existence problems* P-E and N-E.

Definition 5 (VLC existence problems)**.** *Given an ARDA framework $\mathcal{AF} = \langle L, A, D, \mathcal{P}, a \rangle$, P-E($\mathcal{AF}$) (resp., N-E($\mathcal{AF}$)) is the problem of deciding if at least one possible (resp., necessary) VLC w.r.t. a exists.*

Example 2. *Consider the ARDA framework $\mathcal{AF} = \langle \{l_1, l_2\}, \{x_1, x_2, x_3\}, \{(x_2, x_1), (x_3, x_2)\}, \mathcal{P}, x_1 \rangle$, and assume that the analyst is investigating the leakage of x_1. The profiles of the agents are:*

- *$P_{l_1}(x_1) =$ yes, $P_{l_1}(x_2) =$ maybe, $P_{l_1}(x_3) =$ yes, $P_{l_1}((x_2, x_1)) = P_{l_1}((x_3, x_2)) =$ yes*
- *$P_{l_2}(x_1) =$ yes , $P_{l_2}(x_2) =$ yes , $P_{l_2}(x_2, x_1) =$ yes*

In the iAAF induced by the coalition $\{l_1\}$, x_1 and x_3 are certain arguments, while x_2 is uncertain. In the completions where x_2 is present, it attacks x_1 but is itself attacked by x_3. In contrast, in the completions where x_2 is absent, x_1 is not attacked at all. In both cases, x_1 is accepted and therefore $\{l_1\}$ is a necessary VLC. We now consider the iAAF induced by the coalition $\{l_1, l_2\}$. In this case x_3 is no longer a certain argument and, therefore, x_1 is no longer defended in every completion. This makes $\{l_1, l_2\}$ a possible VLC, but not a necessary one. Finally, in the iAAF induced by $\{l_2\}$, x_1 and x_2 are certain arguments and x_1 is attacked by x_2. Consequently, x_1 is not accepted in any completion, and there is no VLC.

Solving the existence problems can be useful at least to support a preliminary analysis. For instance, a *No* answer from p-e($\mathcal{AF}$) provides the analyst with the information that no coalition made of the candidate leakers in L might have found the leaked argument justified; thus, the analyst may decide to invest their effort to see if the list of suspects can be widened, by considering suspects currently not included in L.

The fundamental decision problems related to the search of VLCs and MVLCs are defined as follows.

Definition 6 (VLC and MVLC verification)**.** *Given an ARDA framework $\mathcal{AF} = \langle L, A, D, \mathcal{P}, a\rangle$ and a set of candidate leakers $C \subseteq L$, p-ver($\mathcal{AF}, C$) (resp., p-verMin($\mathcal{AF}, C$)) and n-ver($\mathcal{AF}, C$) (resp., n-verMin($\mathcal{AF}, C$)) are the problems of deciding if C is a possible and a necessary VLC (resp., MVLC) w.r.t. a, respectively.*

We now discuss some properties characterizing the reasoning over an ARDA framework based on the notions of VLC and MVLC, as well as on the fundamental problems defined so far.

Proposition 1. *Let $\mathcal{AF}$ be an ARDA framework and C be a set of candidate leakers.*

1. *If C is a necessary VLC, then C is a possible VLC.*
2. *If C is a necessary MVLC, then C is a possible MVLC.*

Proposition 1 states a rather expected property: what is necessary is also possible. Indeed, while this is straightforward in the case of VLCs, it is not all that trivial for MLVCs. In fact, at a first glance, there might exist a coalition C such that: i. C is $\subseteq$-minimal w.r.t. the property that a is accepted in all the completions of F_C, but also ii. one of the strict subsets C' of C is such that a is accepted in one completion of $F_{C'}$. However, such a counterexample does not exist, but we will discuss this issue later on, as implication 2. will turn out be an immediate consequence of the results of the investigation that we report the following paragraphs, where we provide further formal properties characterizing the reasoning over the ARDA framework.

Lemma 1. *If a coalition of candidate leakers C is a necessary VLC w.r.t a, then every $C' \subset C$ (with $C' \neq \emptyset$) is a necessary VLC w.r.t a.*

The following proposition is an immediate consequence of the above lemma.

Proposition 2. *If C is a necessary MVLC w.r.t. a, then C is a singleton.*

The above proposition means that the guarantees provided by the necessary perspective are so strong that the requisites for being a necessary MVLC are satisfiable at most by coalitions consisting of a single leaker. An immediate consequence of this property is implication 2 of Proposition 1 above: as every necessary MVLC C is a singleton, no strict subset of C can be a VLC, so necessary MVLCs are also possible MVLCs.

We now focus on the possible perspective, and show that, in this case, a behavior that is dual to what is stated in Lemma 1 for the necessary perspective can be observed.

Proposition 3. *If C is possible VLC w.r.t a, then every $C' \supset C$ is possible VLC w.r.t a.*

An immediate consequence of Proposition 3 is the following result.

Corollary 1. *p-e($\mathcal{AF}$) coincides with p-ver($\mathcal{AF}, C$), where $C = L$.*

Thus, as stated in Corollary 1, the monotonicity property stated in Proposition 3 implies that the existence of a possible VLC can be decided by solving an instance of p-ver, where the whole L is considered as the target coalition. Another aspect giving relevance to Proposition 3 is that it suggests strategies for solving p-verMin and for searching an MVLC. As for p-verMin($\mathcal{AF}, C$), Proposition 3 ensures that it can be solved by solving at most $|C|$ instances of p-ver, where the target coalitions are obtained from C by removing one candidate leaker at a time (in fact, the answer of p-verMin is *yes* if and only if the answers of these $|C|$ instances are all *No*). Analogously, Proposition 3 implies that the following strategy for searching an MVLC is correct: starting from $C = L$, iteratively remove from C one of its candidate leakers such that the updated C is an *yes* instance of p-ver, and return the current C when no such candidate leaker can be found.

We will discuss other computational aspects related to problems introduced over the ARDA framework in the next section, when discussing their relationship with classical reasoning problems over iAAFs.

4. Related work and conclusive discussion

Identifying the source of sensitive data leaks is a key challenge in cybersecurity. Two surveys address this issue: [8] highlights techniques using smart documents that record access metadata (e.g., user, time, location), biometric tracking, semantic signatures, and behavioral machine learning to support attribution. [9] explores culprit attribution techniques, such as *fake objects* (used as watermarks within distributed data to trace the source of leaks), probabilistic models, minimal-overlap distributions, and personalized data tracking. In fact, most existing approaches to data leakage attribution are grounded in physical evidence (such as biometric signals or presence tracking) to determine responsibility. In this work, we propose a fundamentally different strategy based on logical reasoning. We propose that the content and logical coherence of leaked information can be treated as evidential clues. Specifically, we consider coalitions of agents who, by aggregating their available knowledge, would be capable of logically inferring the leaked conclusion. This epistemic analysis complements traditional technical approaches by incorporating reasoning capacity as a criterion for suspicion. Our work thus falls within the line of research in which logical frameworks (based on defeasible logic and/or argumentation [10, 11, 12]) are studied to address attribution problems in the field of cybersecurity and, to the best of our knowledge, it is the first to specifically consider the context of data leakage.

Regarding the relationship between ARDA and iAAFs, we observe that P-VER and N-VER can be easily solved by resorting to standard machineries for solving the acceptance tests over iAAFs (in fact, P-VER and N-VER are equivalent to deciding the -possible or necessary- acceptance of an argument over the iAAF induced by the coalition). In turn, given the relationship we highlighted between the existence and the verification problems for VLCs, also P-E and N-E can be solved by exploiting acceptance solvers over iAAFs. Finally, as for the verification of minimal VLCs, it is trivially reducible to the verification of VLCs under the necessary perspective, while under the possible perspective, due to Proposition 3, it can be solved by means of at most $|L|$ invocations of a solver of the acceptance in an iAAF.

References

[1] P. M. Dung, On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games, Artif. Intell. 77 (1995) 321–358.

[2] B. Fazzinga, S. Flesca, F. Furfaro, L. Pontieri, Process mining meets argumentation: Explainable interpretations of low-level event logs via abstract argumentation, Inf. Syst. 107 (2022) 101987.

[3] B. Fazzinga, A. Galassi, P. Torroni, An argumentative dialogue system for COVID-19 vaccine information, in: Proc. Int. Conf. on Logic and Argumentation (CLAR), Springer, 2021, pp. 477–485.

[4] L. Amgoud, H. Prade, Using arguments for making and explaining decisions, Artif. Intell. 173 (2009) 413–436.

[5] T. Bench-Capon, K. Atkinson, A. Wyner, Using argumentation to structure e-participation in policy making, Trans. on Large-Scale Data- and Knowledge-Centered Systems XVIII (2015) 1–29.

[6] D. Baumeister, D. Neugebauer, J. Rothe, H. Schadrack, Verification in incomplete argumentation frameworks, Artif. Intell. 264 (2018) 1–26.

[7] D. Baumeister, M. Järvisalo, D. Neugebauer, A. Niskanen, J. Rothe, Acceptance in incomplete argumentation frameworks, Artif. Intell. 295 (2021) 103470.

[8] I. Herrera Montano, J. J. Garcia Aranda, J. Ramos Diaz, S. Molina Cardin, I. De la Torre Díez, J. J. Rodrigues, Survey of techniques on data leakage protection and methods to address the insider threat, Cluster Computing 25 (2022) 4289–4302.

[9] R. Verma, V. Gautam, C. P. Yadav, I. Gupta, A. K. Singh, A survey on data leakage detection and prevention, in: Proc. Int. Conf. on Innovative Computing & Communications (ICICC), 2020.

[10] A. Applebaum, K. N. Levitt, Z. Li, S. Parsons, J. Rowe, E. Sklar, Cyber reasoning with argumentation: Abstracting from incomplete and contradictory evidence, in: Proc. Military Communications Conference (MILCOM), 2015, pp. 623–628.

[11] E. Karafili, A. C. Kakas, N. I. Spanoudakis, E. C. Lupu, Argumentation-based security for social good, in: Proc. AAAI Fall Symposia, 2017, pp. 164–170.

[12] E. Nunes, P. Shakarian, G. I. Simari, A. Ruef, Argumentation models for cyber attribution, in: Proc. Int. Conf. on Advances in Social Networks Analysis and Mining (ASONAM), 2016, pp. 837–844.

Appendix: proofs of propositions and lemmas

Proposition 1. Let $\mathcal{AF}$ be an ARDA framework and C be a set of candidate leakers.

1. If C is a necessary VLC, then C is a possible VLC.
2. If C is a necessary MVLC, then C is a possible MVLC.

Proof.
1. Trivial.
2. If C is not a possible MLVC, then there exists one of the strict subsets C' of C such that a is accepted in one completion of $F_{C'}$. However, Proposition 2 states that if C is a necessary MVLC, then C is a singleton. Thus, no strict subset of C can be a VLC, so necessary MVLCs are also possible MVLCs. □

Lemma 1. If a coalition of candidate leakers C is a necessary VLC w.r.t a, then every $C' \subset C$ (with $C' \neq \emptyset$) is a necessary VLC w.r.t a.

Proof. Moving from C to C' means moving the reasoning from over F_C to over $F_{C'}$. Since $C' \subset C$ and $C' \neq \emptyset$, $F_{C'}$ can be viewed as the result of the following two tasks:
1) removing from F_C some uncertain arguments/attacks, i.e. every argument/attack x such that there is an agent $l \in C$ with $P_l(x) \in \{$ *yes, maybe* $\}$ but no agent l' in C' with $P_{l'}(x) \in \{$ *yes, maybe* $\}$,
2) making certain some uncertain arguments and attacks of F_C, i.e. the arguments and attacks such that $\forall l' \in C'\ P_{l'}(x) =$ *yes* while $\exists l \in C$ such that $P_l(x) =$ *maybe* or P_l is not defined over x.
Hence, the set of completions of $F_{C'}$ is a subset of the set of completions of F_C, and this implies that every argument necessarily accepted in F_C is also necessarily accepted in $F_{C'}$. □

Proposition 2. If C is a necessary MVLC w.r.t. a, then C is a singleton.

Proof. Lemma 1 implies that no necessary VLC containing two or more candidate leakers can be minimal. □

Proposition 3. If C is possible VLC w.r.t a, then every $C' \supset C$ is possible VLC w.r.t a.

Proof. As $C' \supset C$, $F_{C'}$ can be viewed as the result of: 1) adding new uncertain arguments and/or attacks to the F_C, and/or 2) making uncertain some certain arguments and/or attacks already occurring in F_C. Therefore, the completions of F_C are still completions of $F_{C'}$, so any completion of C where a is accepted is also a completion of $F_{C'}$. □

Corollary 1. P-E($\mathcal{AF}$) coincides with P-VER($\mathcal{AF}, C$), where $C = L$.

Proof. If the answer of P-VER($\mathcal{AF}, C$), where $C = L$, is *yes*, then the answer of P-E($\mathcal{AF}$) is obviously *yes* as well. Vice versa, if there is a VLC C, then Proposition 3 implies that also L is a VLC, since $L \supseteq C$. □

Discovering the Potential of LLMs in Annotating Legal Texts for Argument Mining

Christina Berghegger[1,†], César Philippe[1,†], Karla Salas-Jimenez[1,2,†], Jean-Guy Mailly[1,†], Leila Moudjari[3] and Laurent Perrussel[1]

[1]Université Toulouse Capitole, IRIT, Toulouse, France
[2]PCIC, UNAM, Ciudad de México, México.
[3]IRIT, Université de Toulouse, CNRS, Toulouse INP, UT3, Toulouse, France

Abstract

Legal reasoning is a key domain for the application of computational argumentation techniques. However, the scarcity of datasets of annotated legal text tailored for argumentation-related tasks—aside from a single dataset comprising decisions from the European Court of Human Rights (ECHR)—poses a significant challenge to developing automated legal reasoning systems grounded in formal argumentation techniques. To address this challenge, we explore the use of Large Language Models (LLMs) to facilitate the annotation of legal texts, specifically focusing on identifying premises and claims within the ECHR dataset. Our preliminary results demonstrate the promising potential of LLMs to assist legal experts in annotating additional legal corpora. This advancement paves the way for more robust, argumentation-driven automated reasoning systems in the legal domain.

Keywords

Argument Mining, Legal Argumentation, Dataset annotation, ECHR, LLM

1. Introduction

Computational models of argumentation [1, 2] have shown their usefulness in modeling various kinds of real world situations, including legal reasoning. A key part of legal reasoning is based on the use of arguments, *e.g.,* to convince a jury about the innocence or guilt of a defendant or to justify the decision of a court. The resolution of a case depends on the arguments provided by each party's theory of the case. In other words, the arguments are the cornerstone of the law and developing tools to extract these arguments would help litigants greatly. It will help improve the definition of the case, find contradictions made by the other party or provide a reasoned and motivated resolution. An illustrative and recent example of an application of computational argumentation for legal reasoning is the use of the ANGELIC method [3] to predict the decisions of the European Court of Human Rights (ECHR) [4], showing that this kind of technique based on symbolic reasoning has better accuracy than approaches purely based on Machine Learning.

The practical application of argument-based reasoning to legal scenarios relies on the extraction of arguments and their relations from a variety of legal text. Such an extraction requires the development of efficient argument mining techniques [5], but a weak point in the current state of the art in legal argument mining is the lack of annotated datasets.

The recent emergence of LLMs and their success for various tasks [6, 7] offer opportunities to improve the extraction of arguments from legal text. Various LLM tasks have been performed on the ECHR dataset, such as sentence classification [8], prompt engineering [9], or argument extraction [10].The fact that there is no comparable dataset in the legal field creates little room to test generalizability or compare results on other baselines.

Arg&App 2025: International Workshop on Argumentation and Applications, November 2025, Melbourne, Australia
[†]These authors were funded by the French National Research Agency (grant AIDAL, ANR-22-CPJ1-0061-01).
christina.berghegger@ut-capitole.fr (C. Berghegger); cephilip@insa-toulouse.fr (C. Philippe); karla_dsj@ciencias.unam.mx (K. Salas-Jimenez); jean-guy.mailly@irit.fr (J. Mailly); leila.moudjari@irit.fr (L. Moudjari); laurent.perrussel@ut-capitole.fr (L. Perrussel)


The goal of this paper is to demonstrate how LLM can facilitate the annotation of text by legal experts. As in Gray et al. [11], we will use LLM for determining whether a sentence in a legal opinion describes a legal factor. Our main contributions are as follows:

- **Novel methodology for LLM as a legal annotator**: We propose a methodology that addresses a critical challenge in legal NLP. It is based on several LLM prompts that could help with daily tasks for legal practitioners.
- **Structured Evaluation Pipeline**: We design and compare multiple annotation pipelines (Direct, Direct with Complaints, and Modular), demonstrating thoughtful engineering and methodological rigor.
- **In-depth Evaluation**: We conduct both quantitative (similarity thresholds, match ratios) and qualitative (manual annotation reviews, iterative prompt refinement) evaluations to assess performance and guide improvements.

The paper is structured as follows: Section 2 explores the usage of LLMs for applications in the field of computational argumentation. In Section 3, we detail the setup of our annotation pipeline[1]. The experiments performed with different models and the results are explained in Section 4. Finally, we sketch out some future ideas for our project in Section 5.

2. Related Work

As most fields in Natural Language Processing (NLP), natural argumentation has been heavily impacted by the emergence of LLM. Researchers started evaluating the performance of LLMs in well-known argumentation tasks, such as premise and claim identification [12] or argument scheme classification [13], but also on new use cases, such as interactions between LLM agents [14]. The present work focuses on LLMs for data annotation for argument mining.

LLMs to annotate data One of the most relevant is the work of Trajano et al. [15], they observe that it can be challenging to find experts available to evaluate the correctness of the LLM results. Next, Lindahl and Borin [16] compare the inter-annotator agreement (IAA) for different texts, and state that a "less than substantial agreement is a quite common result in argumentation mining" due to the lack of a unified or generally accepted argumentation model and the fact that approaches to argumentation differ depending on the domain type. Pietron et al. [17] observe "a noticeable group of incorrect or problematic classifications made by human evaluators". Savelka and Ashley [18] explore the ability of LLMs to semantically annotate legal texts in a zero-shot learning settings, which is a relevant research question in fields where either little data for finetuning is available, or finetuning is computationally not possible. They consider tasks such as contract review or case analysis, and compare the performance of different models, namely on semantic annotation tasks.

LLMs for argument mining There are papers that use LLMs for argument mining. For example, Trajano et al. [15] explore an approach to translating natural language arguments into computational arguments using LLMs with an argumentation scheme provided from Retrieval Augmented Generation (RAG). Zubaer et al. [19] compare the performance of GPT-like models in legal argument mining via prompting between zero-shot and with some examples, specifically in clause classification, using the legal corpus from the European Court of Human Rights (ECHR). They explore in-context learning (ICL), which involves guiding a language model using instructions and examples rather than fine-tuning. They highlight that prompts act like hyperparameters needing careful tuning and find that providing examples generally improves performance over zero-shot approaches. De Wynter and Yuan [9] evaluate the abilities of LLMs in argument mining and argument pair extraction, considering different levels of abstraction in their input and output. Gorur et al. [20] shows the interest of LLMs for identifying the

[1]Source code and complete prompts available at https://github.com/KarlaDSJ/LegalArgumentsLLM

attacks/supports relations between arguments. The approach outperforms state-of-the-art methods, but requires that the arguments have already been correctly identified. Faugier et al. [14] propose an LLM-based debate builder aiming at verifying the existing relations between arguments in a debate and assisting users to create new arguments, namely of type support and attack.

Moreover, although computational argumentation is a highly interdisciplinary field in which many tasks can be applied to different domains, each domain has particularities that must be considered. For illustration, Otiefy and Alhamzeh [12] explore argument mining tasks such as claim/premise and argument relation classification in the financial field, and Hong et al. [21] propose an agent-based framework based on LLM discussions via argumentation schemes, aiming at explainable clinical decision reasoning.

The work by Mumford et al. [22], presenting a new annotated dataset covering Article 6 of the European Convention on Human Rights (ECHR), shows the extensive effort which has to be undertaken to create new annotations. This motivated us to take a step backwards and, before proceeding to argument mining tasks, have a closer look at the potential of data annotation with the help of Large Language Models. We want to explore the possibility to annotate datasets for a variety of tasks, and in a variety of domains, minimizing time and costs. Notice that our goal is not to extract arguments with LLMs (*i.e.* to perform the argument mining task with LLMs), but to assist annotators in their task to created new datasets. The identification of additional arguments is an implicit aspect of this support, not the main focus. The dataset published by Poudyal et al. [23] is commonly used for legal argumentation tasks, and will serve as a baseline for our work.

3. Argumentation Pipeline

Let us now detail the pipeline for extracting arguments for annotations: in a first step, we detail the input, *i.e.* the dataset and second we describe the technical configuration.

3.1. Selected cases

Short texts					Long texts				
text	**length**	**words**	**tokens**	**ratio**	**text**	**length**	**words**	**tokens**	**ratio**
00	39	1789	2315	29%	01	127	2002	2381	19%
04	41	978	1281	31%	02	158	1462	2118	45%
05	24	993	1242	25%	17	175	2002	2739	37%
06	26	1130	1435	27%	19	130	2002	2505	25%
10	32	2002	2456	23%	27	128	2002	2544	27%
13	27	851	1068	25%	29	118	2002	2613	31%
16	32	2002	2537	27%					
20	54	2002	2465	23%					
21	70	2002	2513	26%					
mean	38.3	1527.7	1923.6	26%		139.3	1912	2483.3	31%

Table 1
The lengths of the texts are presented in terms of the number of lines, words, tokens, and ratios (ratio stands for 'word to token ratio' because in the tokenizer, a word is not necessarily one token).

As mentioned in previous sections, the only annotated dataset of legal texts (for argument mining purposes) is the one created by Poudyal et al. [23], the ECHR dataset. It is composed of 42 decisions of the European Court of Human Rights (ECHR). It is annotated by premise, conclusion, and non-argument. Recently, Habernal et al. [10] proposed an extended version of the ECHR dataset with token span labels. However, this new dataset only annotates arguments, not the parts of arguments (claims and premises).

As a baseline, we identified nine texts of the ECHR dataset with rather short text length. We chose short texts as input during the development process of the pipeline to accelerate testing and to rapidly identify possible shortcomings. In order to test the generalization of the pipelines in longer text we

identified six longer texts from the ECHR dataset. Table 1 details the characteristics of the short and long text used in this work.

3.2. Technical Setup

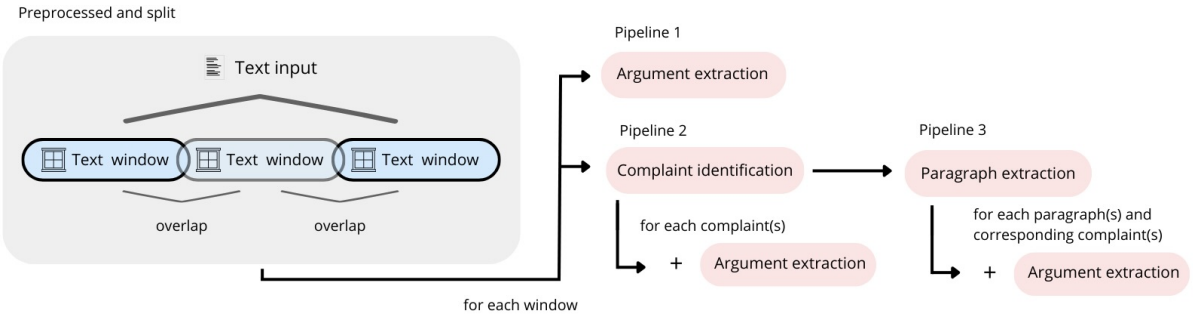


Figure 1: Schematic description of the three variants of our pipeline, where the text input is split in several text windows, and then either (1) the LLM is used to extract arguments directly, or (2) the complaints in the windows are identified before the argument extraction, or (3) for each complaint, there is a paragraph extraction before argument extraction.

To capture the idea of modularity in terms of task execution, we initiate our project with three different types of pipelines, each having a different degree of modularity (see Figure 1). All pipelines have two versions, one of whose prompts are extended by examples, inspired by the literature around In-Context Learning as mentioned in Section 2.

Pipeline 1: *Direct from text* retrieve arguments directly from a legal text, without any intermediary tasks, pre- or post-processing. This is the least modular pipeline.

Pipeline 2: *Direct with complaints* First, we identify the applicant's complaints. Then, the LLM retrieves arguments from the original text that address the complaints.

Pipeline 3: *Modular* The idea of this pipeline is to replicate a behaviour which we consider to be a logical way of human reasoning: First, we search for relevant topics (complaints), then we identify their contexts (paragraphs) and finally draw conclusions (arguments).

Technical Configuration We used two models for comparison: the *meta-llama/Meta-Llama-3-8B-Instruct* [24], which has eight billion parameters, and the specialized law model *Equall/Saul-7B-Instruct-v1* [25] based on Mistral, which has seven billion parameters and implements a continuous pretraining using an English legal corpus of over 30 billion tokens from various jurisdictions (in the U.S., Europe and Australia). We set the temperature to 0.5 and the maximum number of output tokens to 1024.

We limited output tokens due to the 2000-word maximum output length of current LLMs, as shown by Bai et al. [26]. Since we lacked access to Meta-*Meta-Llama-3-8B-Instruct* to determine its word-token ratio, we used the tokenizer from *sentence-transformers/all-MiniLM-L6-v2* to estimate the ratio based on our input texts.

We analyzed the first 2000 words of each input text and found an average word-token ratio of 29%, equivalent to 2576 tokens (see Table 1). For now, we decided to limit the output pipeline to a far smaller number to account for the possible usage of models with smaller ratios. However, especially when working with longer texts as presented in subsection 4.2, testing larger output threshold should be explored as a way to improve pipeline performance. Given that we limit our output size, we also decided to split the input text into smaller windows. This has advantages and disadvantages. On the one hand, in case the entire text window is relevant and composed of arguments, the output size is large enough to retrieve all relevant information. On the other hand, the LLM might lose context, because it can only

refer to information from one text window at a time. We try to limit the latter by ensuring an overlap between text windows (see Figure 1).

Annotation scheme. For each pipeline, we provide a prompt and a guideline for annotating every argument found in it. We expect granular annotations of each argument, identifying the claims and premises. Guideline example:

```
n. Argument
**Claim:** {text of the claim}
**Premises:**
- {premise 1}
- {premise 2}
...
```

3.3. Pipelines Evaluation

We evaluated the LLM generated arguments in terms of their similarity with original annotations from the ECHR dataset. We calculate the similarity using the tokenizer and embeddings of the *sentence-transformers/all-MiniLM-L6-v2* model. Using vector similarity as comparison metric, we give some more freedom to the LLM, and introduce some degree of flexibility regarding the identification and interpretation of arguments. This is in line with the low annotators' agreement we could observe from many related studies, as presented in Section 2, and which suggests that argument annotations in the legal context are seldom straightforward.

We proposed two ways for evaluating the similarity of an argument: 1) Evaluating whole argument, without taking into account their internal structure, and 2) Evaluating every part of the argument, in a structured way. For example:

a) Unstructured argument

Argument: The Government submit that the Magistrates made an error of law within their jurisdiction which has been corrected on appeal. They rely on the absence of any suggestion in the High Court's judgment that the applicant's detention was unlawful.

b) Structured argument

Claim: The Government submit that the Magistrates made an error of law within their jurisdiction which has been corrected on appeal.

Premises: They rely on the absence of any suggestion in the High Court's judgment that the applicant's detention was unlawful.

Unstructured argument We use vector similarity as our comparison metric. The similarity is computed in two directions: 1) from each argument in the original dataset to the argument generated by the LLM (*base on original*), and 2) from each argument generated by the LLM to the arguments in the original dataset (*base on LLM*). An argument a and an argument b are considered a match if their vector similarity (denoted by Function $sim(a, b)$) exceeds a predefined threshold t. For this work, we set $t = 0.75$. Therefore, the *matching ratio* indicates the percentage of arguments that matched the process described above. In Appendix A, we present experiments for setting value t.

Structured argument We take the matched arguments for the *Unstructured argument*, computed the similarity between each claim and premises and evaluate the granular similarity by assigning different weights to premise and claim similarity: for $\alpha \in]0, 1[$, $sim\_arg(A_1, A_2) = \alpha \times simP(P_1, P_2) + (1 - \alpha) \times simC(C_1, C_2)$ with A_i an argument made of premises P_i and a claim C_i, and $simP$ (resp. $simC$) the function evaluating the similarity between premises (resp. claims) [27]. In our case, these functions are again the vector similarity, so the main point is to determine the relative importance of premises and claims (*i.e.* the value of α). We ran several experiments to determine the value of α as 0.7. The experiments are presented in Appendix B.

4. Experiments and results

In this section, we first present a comparison of the three pipelines with zero-shot and one-shot prompts. Then, we conduct an in-depth study of Pipeline 1 with a one-shot and a rework prompt, using Llama [24] and Saul [25], and a state-of-the-art Argument Mining methodology (namely ArgueMapper/ArgueBuf) [28, 29] to test similarity in structured and unstructured arguments. Finally, we test the generalisation using long texts.

4.1. Comparison of the 3 pipelines on short texts

We ran all the pipelines on Llama [24], with and without examples, for the short texts, and did a high-level comparison with the original annotations. We took into account the mean number of arguments per text, the mean number of premises per argument, as well as the maximum number of premises belonging to one argument. As presented in Table 2, the pipeline *Direct from text with examples* is the closest to the original, and was thus used to generate outputs for all following evaluations and analysis. While the high-level evaluation gives a first indication that the pipeline with examples tends to output less arguments with more premises compared to its zero-shot setting, we want to get a more detailed understanding of the impact of providing examples. Hence, we keep the zero-shot pipeline to compare annotation results.

Pipeline	mean #Arg.	mean #Prem.	max #Prem.
Original dataset	**13.56**	**2.49**	**5.56**
Direct from text (−)	25.22	2.01	**4.56**
Direct from text (+)	**16.44**	**2.21**	**4.56**
Direct with complaints (−)	25.56	1.99	4.22
Direct with complaints (+)	16.56	1.91	4.33
Modular (−)	23.22	1.82	4.44
Modular (+)	19.89	1.57	3.89

Table 2
High-level pipeline evaluation on mean number of arguments, and mean and max number of premises per arguments, for the original dataset and the three pipelines with (+) or without (−) examples

Let us focus on Pipeline 1 *Direct from Text* and analyse the impact of providing examples to the LLM on annotation performance by evaluating the matching ratio, as well as similarity scores. We observed that the zero-shot pipeline extracted a lot more arguments than present in the original annotations. As shown in Table 3 column *zero-shot*, the matching ratio is lower on LLM arguments for the zero-shot setting.

Based on	zero-shot
Original	70.69%
LLM	55.16%

Table 3
Matching ratio 0.75 for Pipeline 1 Direct from Text with zero-shot. The column *Base on* refers to the base data set for calculating similarity.

The fact that the matching ratio based on original arguments is higher for the zero-shot setting indicates a low precision of LLM generated arguments. In other words, the LLM extracts too many arguments which increases chances of one of them being a good match for an original argument, but leaving a higher number of extracted arguments unmatched, decreasing the precision score. Besides this, the quality of matches is rather equal for the prompt with examples and the zero-shot setting, with a slightly better performance for the prompt with examples. Mean values over all texts are provided in Table 4.

Metric	Based on original		Based on LLM	
	examples	zero-shot	examples	zero-shot
mean sim	83.22%	82.58%	83.33%	82.49%
max sim	85.30%	85.05%	85.46%	84.05%

Table 4
The mean and maximum similarity are calculated after matching the arguments based on the original and those generated by the LLM, both with and without examples.

Recall that our aim is to create a pipeline which generates annotations on legal texts. Hence, precision is of high importance and a pipeline which just creates an excessive amount of arguments, which however are not relevant, cannot be used. Therefore, we still believe that the pipeline using examples, and extracting arguments with a higher precision, is the best one to go forward with. Hence, the following sections are based on the setting with examples.

4.2. Improving the Generation of Annotations

Aiming to increase quality of the generated annotations, we adopt an iterative prompt modification approach, targeting different weaknesses of a pipeline. For example, to increase the ratio of matched original arguments, it is important to understand the reasons why a specific argument of the original annotations cannot be identified by the LLM. Therefore, as a first iteration step, we did a manual evaluation of all non-matched original arguments and identified the following recurring characteristics:

1. The LLM does not identify arguments of complex structures. It is good at identifying arguments where the premises indicates in an explicit manner the support of a claim, but it cannot capture situations where this support is less evident.
2. LLM arguments are often too short, *e.g.* composed of only a claim and lacking premises completely.
3. The LLM missed several arguments where references were made to prior cases or legal articles.

We changed the LLM prompt accordingly and added the following instructions:

1. We provided specific structures and syntax arguments can have and explained the respective relations between premises and claims. For this iteration, we used three schemes that are widely discussed in literature, *e.g.* argument schemes around position to know, rules and precedent cases which we found relevant for our application [30].
2. We gave specific instructions about the requirement of premises, and indicated a preference for longer arguments.
3. We explained the role of prior cases and references to articles, and gave specific instructions to use them to form arguments.

After modifying the prompt, we ran the pipeline again on Llama [24] and re-evaluated the match ratio. Let us illustrate the impact of this change on Pipeline 1: *Direct from text*: we could observe that it increases for the values of the original arguments (cf. Table 5 columns *before* and *after*), which is an indication for a positive impact of our modifications to match original arguments.

This can be explained by the fact that our first iteration of prompt modifications was targeting unmatched original arguments, and unmatched LLM arguments were not taken into account when altering the prompt. In the following prompt improvement iterations, we should thus consider unmatched LLM arguments to increase precision.

Long text Similarly to the shorter texts, we ran the pipeline on Llama [24] with the initial and modified prompt for long texts. As we can see in Table 5, the mean matching ratio looks generally better for long texts, than for shorter texts. It must be noted that the aforementioned matching ratio on LLM arguments is a lot worse for the long text inputs. This means, that the LLM generated a lot more arguments than annotated in the original dataset, and while many original arguments can be matched, the majority of LLM arguments does not appear in the original annotations. This is another motivation to look at matching ratios from both perspectives.

Limitation Figure 2 details the results for each text. Improvement cannot always be guaranteed: when comparing values *before* with values *after*, the similarity of arguments decreased for some texts (namely texts 00, 13 and 20). It requires a more fine-grained analysis during the prompt improvement iteration. Moreover, the modifications did not lead to an equal increase in the precision score of LLM extracted arguments, where the ratio stayed rather stable as presented in Table 5 columns *before* and *after*.

Based on	zero-shot	before	after	long-texts before	long-texts after	Saul
Original	70.69%	64.85%	76.40%	70.79%	84.81%	71.04%
LLM	55.16%	62.44%	61.92%	41.01%	46.47%	72.06%

Table 5
Mean matching ratio 0.75 for pipeline 1. We show the value for all prompt modifications. The column *Base on* refers to the base data set for calculating similarity.

Towards an interactive annotation We believe that this iterative prompt improvement is a really interesting perspective to understand strengths and weaknesses of our pipeline. In a real system, these feedback loops could be part of the pipeline, allowing for output improvement through human-computer interaction. Alternatively, the interaction could also be automatized and completed by multiple LLM-agents.

Considering the structure of the arguments We computed the granular similarity for the structured arguments, as can be seen in Figure 2 in the graph on the right. For all the documents, the similarity decreased significantly. In a detailed analysis, we identified four main reasons for this:

- The Argumentative Discourse Units (ADUs) are the same, but they are not ordered. The claim is a premise, and one premise is the claim. The rest of the premises are also correct premises.
- The same ADUs, but with more information. The premises found by the LLM are in the original argument, but the claim did not appear.
- The ADUs are disorganized and missing information. The claim found is a premise, and one premise is the original claim, but the LLM did not find some premises.
- The ADUs are partially found. The claim was not found, and the premises coincide partially.

We list the cases in order of difficulty, with the easiest ones at the top. We prefer to have the entire argument, even if it is somewhat disorganised. This makes it easier for a lawyer to organise the argument than to search for missing information. These points could be interesting to take into account for Pipeline 2.

4.3. Comparison with Other Approaches

To understand the impact of the LLM model on results, we ran our pipeline with another LLM: *Equall/Saul-7B-Instruct-v1* [25].

As presented in Figure 2, the orange bar indicates the results of the reworked prompt. As can be seen, the performance of Saul is lower than that of Llama. This is not surprising because Saul is smaller than Llama. The advantage is that Saul produces fewer arguments, as can be seen in Table 5.

We also evaluated a state-of-the-art argument mining approach (namely ArgueMapper/ArgueBuf [28, 29]) to identify arguments in our dataset. The pipeline is designed to transform texts into structured arguments and comprises stages to extract linguistic features, classify premises and claims, and associate premises with their corresponding claim. It works with classic NLP (Natural Language Processing) and ML (Machine Learning) algorithms. We trained the model on all texts from the dataset except for the 9 short texts for which we generated annotations and see that the model outperforms the LLM approaches.

For this algorithm, we achieved better results in terms of both similarity and granular similarity for the documents: 06, 13 and 16. For the rest of the documents, the performance is similar to that of Llama. Regarding granular similarity, there are cases where the algorithm only finds a claim without premises.

Although the results look promising, we must again mention our goal of annotating data from scratch, with a versatile, modular, and independent pipeline. Approaches like these need already annotated data to function properly, which can be a drawback for our use case, despite the high performance. Moreover, given that train and test data belong to the same dataset, we would also have to test generalizability of the model on other datasets.

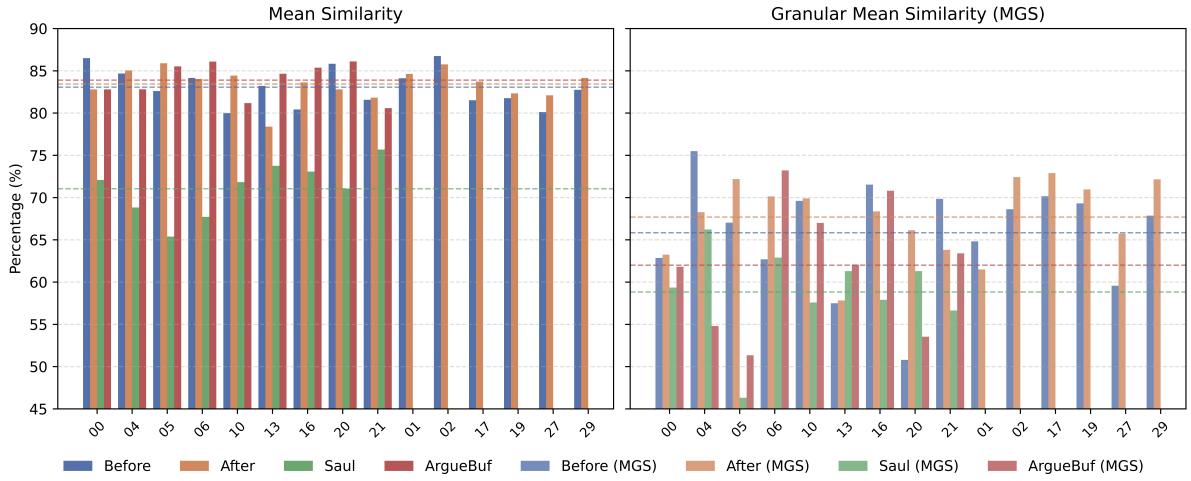


Figure 2: The mean and granular similarities are presented. The first nine texts are the shortest texts, compared with other approaches, and the last six are a generalization on Llama for long texts.

5. Conclusion

The emergence of LLMs has introduced new possibilities to the field of computational argumentation. However, there are still challenges to overcome, such as the limited availability of data, the difficulty of finding experienced evaluators, and the need for task-specific annotations, which are often time-consuming and expensive to produce. To address these issues, we explored the potential of LLMs for creating annotations.

We propose a pipeline based on the ECHR dataset and use prompting with examples to extract arguments from the original texts. We then compare the LLM-extracted arguments to the ground truth based on vector similarity of arguments and use the matching ratio, *i.e.* the number of original arguments that were identified by the LLM, as performance metric. We observe that the ratio fluctuates significantly between texts, but reaches a mean score of more than 64% for the initial data input and prompt. After prompt improvement, this ratio increases by more than 11 percentage points to over 76%. This project creates a baseline which we believe can be significantly improved by extending the pipeline. For example, we can think of an automated prompt improvement process where LLM-agents challenge and evaluate each other aiming at reaching as much similarity with the ground truth as possible.

For future work, the first step we want to explore is evaluating the efficiency of our pipeline as a tool for legal experts to annotate legal texts. We want to compare the results of two groups of annotators: one that uses the pipeline and another that does not. This comparison will reveal the capabilities of LLMs for annotation tasks, and assess the time saved for the annotation by using the pipeline. In a later step we aim to evaluate the versatility of our pipeline beyond the ECHR dataset, increase its modularity, and generalise it to different tasks. This will make it useful for real-life applications.

References

[1] P. M. Dung, On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games, Artif. Intell. 77 (1995) 321–358. doi:10.1016/0004-3702(94)00041-X.

[2] P. Baroni, D. Gabbay, M. Giacomin, L. van der Torre (Eds.), Handbook of Formal Argumentation, College Publications, 2018. URL: https://www.collegepublications.co.uk/handbooks/?00003.

[3] K. Atkinson, T. J. M. Bench-Capon, ANGELIC II: an improved methodology for representing legal domain knowledge, in: Proc. of ICAIL 2023, 2023, pp. 12–21. doi:10.1145/3594536.3595137.

[4] J. Collenette, K. Atkinson, T. J. M. Bench-Capon, Explainable AI tools for legal reasoning about cases: A study on the european court of human rights, Artif. Intell. 317 (2023) 103861. doi:10.1016/J.ARTINT.2023.103861.

[5] E. Cabrio, S. Villata, Five years of argument mining: a data-driven analysis, in: Proc. of IJCAI 2018, 2018, pp. 5427–5433. doi:10.24963/IJCAI.2018/766.

[6] J. Kaddour, J. Harris, M. Mozes, H. Bradley, R. Raileanu, R. McHardy, Challenges and applications of large language models, CoRR abs/2307.10169 (2023). doi:10.48550/ARXIV.2307.10169.

[7] M. A. K. Raiaan, M. S. H. Mukta, K. Fatema, N. M. Fahad, S. Sakib, M. M. J. Mim, J. Ahmad, M. E. Ali, S. Azam, A review on large language models: Architectures, applications, taxonomies, open issues and challenges, IEEE Access 12 (2024) 26839–26874. doi:10.1109/ACCESS.2024.3365742.

[8] I. Chalkidis, M. Fergadiotis, D. Tsarapatsanis, N. Aletras, I. Androutsopoulos, P. Malakasiotis, Paragraph-level rationale extraction through regularization: A case study on european court of human rights cases (2021) 226–241. doi:10.18653/V1/2021.NAACL-MAIN.22.

[9] A. De Wynter, T. Yuan, "I'd like to have an argument, please": Argumentative reasoning in large language models, in: Proc. of COMMA, 2024, pp. 73–84.

[10] I. Habernal, D. Faber, N. Recchia, S. Bretthauer, I. Gurevych, I. Spiecker genannt Döhmann, C. Burchard, Mining legal arguments in court decisions, Artif. Intell. Law 32 (2024) 1–38. doi:10.1007/S10506-023-09361-Y.

[11] M. A. Gray, J. Savelka, W. M. Oliver, K. D. Ashley, Can GPT alleviate the burden of annotation?, in: Proc. of JURIX 2023, 2023, pp. 157–166. doi:10.3233/FAIA230961.

[12] Y. Otiefy, A. Alhamzeh, Exploring large language models in financial argument relation identification, in: Proc. of FinNLP-KDF-ECONLP 2024, 2024, pp. 119–129. URL: https://aclanthology.org/2024.finnlp-1.12/.

[13] T. Bench-Capon, K. Atkinson, Using argumentation schemes to model legal reasoning, arXiv preprint arXiv:2210.00315 (2022).

[14] E. Faugier, F. Armetta, A. Bonifati, B. Yun, Assisted debate builder with large language models, in: Proc. of ECAI 2024, 2024, pp. 4447–4450. doi:10.3233/FAIA241026.

[15] G. Trajano, D. C. Engelmann, R. H. Bordini, S. Sarkadi, J. Mumford, A. R. Panisson, Translating natural language arguments to computational arguments using LLMs, in: Proc. of COMMA 2024, 2024, pp. 289–300. doi:10.3233/FAIA240329.

[16] A. Lindahl, L. Borin, Annotation for computational argumentation analysis: Issues and perspectives, Lang. Linguistics Compass 18 (2024). doi:10.1111/LNC3.12505.

[17] M. Pietron, R. Olszowski, J. Gomulka, Efficient argument classification with compact language models and ChatGPT-4 refinements, in: Proc. of ICCCI 2024, 2024, pp. 249–262. doi:10.1007/978-3-031-70816-9_20.

[18] J. Savelka, K. D. Ashley, The unreasonable effectiveness of large language models in zero-shot semantic annotation of legal texts, Frontiers Artif. Intell. 6 (2023). doi:10.3389/FRAI.2023.1279794.

[19] A. A. Zubaer, M. Granitzer, J. Mitrovic, Performance analysis of large language models in the domain of legal argument mining, Frontiers Artif. Intell. 6 (2023). doi:10.3389/FRAI.2023.1278796.

[20] D. Gorur, A. Rago, F. Toni, Can large language models perform relation-based argument mining?, in: Proc. of COLING 2025, 2025, pp. 8518–8534. URL: https://aclanthology.org/2025.coling-main.569/.

[21] S. Hong, L. Xiao, X. Zhang, J. Chen, ArgMed-agents: Explainable clinical decision reasoning

with llm disscusion via argumentation schemes, 2024. URL: https://arxiv.org/abs/2403.06294. arXiv:2403.06294.
[22] J. Mumford, K. Atkinson, T. J. M. Bench-Capon, Annotated insights into legal reasoning: A dataset of article 6 ECHR cases, Argument Comput. 15 (2024) 113–119. doi:10.3233/AAC-240002.
[23] P. Poudyal, J. Savelka, A. Ieven, M. F. Moens, T. Goncalves, P. Quaresma, ECHR: Legal corpus for argument mining, in: Proceedings of the 7th Workshop on Argument Mining, 2020, pp. 67–75. URL: https://aclanthology.org/2020.argmining-1.8/.
[24] AI@Meta, Llama 3 model card, 2024. URL: https://github.com/meta-llama/llama3/blob/main/MODEL_CARD.md.
[25] P. Colombo, T. P. Pires, M. Boudiaf, D. Culver, R. Melo, C. Corro, A. F. T. Martins, F. Esposito, V. L. Raposo, S. Morgado, M. Desa, Saullm-7b: A pioneering large language model for law, 2024. URL: https://arxiv.org/abs/2403.03883.
[26] Y. Bai, J. Zhang, X. Lv, L. Zheng, S. Zhu, L. Hou, Y. Dong, J. Tang, J. Li, Longwriter: Unleashing 10,000+ word generation from long context llms, in: Proc. of ICLR 2025, 2025. URL: https://openreview.net/forum?id=kQ5s9Yh0WI.
[27] L. Amgoud, V. David, Measuring similarity between logical arguments, in: Proc. of KR 2018, 2018, pp. 98–107. URL: https://aaai.org/ocs/index.php/KR/KR18/paper/view/17999.
[28] M. Lenz, P. Sahitaj, S. Kallenberg, C. Coors, L. Dumani, R. Schenkel, R. Bergmann, Towards an argument mining pipeline transforming texts to argument graphs, in: Proc. of COMMA 2020, 2020, pp. 263–270. doi:10.3233/FAIA200510.
[29] M. Lenz, R. Bergmann, User-centric argument mining with ArgueMapper and Arguebuf, in: Proc. of COMMA 2022, 2022, pp. 367–368. doi:10.3233/FAIA220176.
[30] T. F. Gordon, D. Walton, Legal reasoning with argumentation schemes, in: Proc. of ICAIL 2009, ACM, 2009, pp. 137–146. doi:10.1145/1568234.1568250.

A. Defining the value of *t*.

To define the value of *t*, we did a manual evaluation of all possible combinations between original arguments and LLM generated arguments of the nine selected articles, and classified them as either *acceptable* or *not acceptable*, without knowing their vector similarity. We obtained the results presented in Table 6. A threshold below 0.6 leads to mostly wrong matches, while matches above 0.85 are almost exclusively correct. For this project, we proceeded with a *t*-value of 0.75, where the majority of matches can be considered acceptable.

Figure 3 and Figure 4 show examples of matches between LLM generated arguments and original arguments from the dataset, with a similarity of 98.4% and 75.1% respectively. The first LLM argument is an almost complete replication of the original, with only one subordinate clause (underlined) missing. The second argument match, which is just above the threshold, shows how the LLM is able to capture the relevant information (exhaustion of domestic remedies, reference to article 26 of the Convention, lodging of the application within a certain time frame) but is not able to replicate the wording and setup of the original argument.

t	acceptable	not acceptable	ratio
0.60	29	97	23%
0.65	26	56	32%
0.70	26	63	29%
0.75	32	30	52%
0.80	37	22	63%
0.85	24	3	89%
0.90	12	1	92%

Table 6
Argument Evaluation per Threshold

The Commission notes that the applicant was detained after having been sentenced by the first instance court to 18 months' imprisonment.
He was released after the Court of Appeal reviewed this sentence, reducing it to 15 months' imprisonment, convertible to a fine.
The Commission finds that the applicant was deprived of his liberty "after conviction by a competent court" within the meaning of Article 5 para. 1 (a) (Art. 5-1-a) of the Convention.

(a) Original argument

The Commission finds that the applicant was deprived of his liberty "after conviction by a competent court" within the meaning of Article 5 para. 1 (a) (Art. 5-1-a) of the Convention.
The applicant was detained after having been sentenced by the first instance court to 18 months' imprisonment.
He was released after the Court of Appeal reviewed this sentence, reducing it to 15 months' imprisonment, convertible to a fine.

(b) LLM generated argument

Figure 3: Original (3a) and LLM generated (3b) arguments with a similarity of 98.4%. Differences are underlined.

the second requirement of Article 26 (Art. 26) of the Convention, namely that of lodging the application within six months of the final effective decision or, if none, within six months of the events of which complaints are made.
The present application was lodged on 3 May 1993, i.e. within the six month period.
He did indeed comply with the six months' rule laid down in Article 26 (Art. 26) of the Convention.

(a) Original argument

The applicant may be said to have complied with the domestic remedies' rule laid down in Article 26 of the Convention.
The applicant exhausted domestic remedies.
The applicant lodged the application within six months of the final effective decision or, if none, within six months of the events of which complaints are made.

(b) LLM generated argument

Figure 4: Original (4a) and LLM generated (4b) arguments with a similarity of 75.1%. Matches of relevant information are underlined.

B. Defining the value of α.

To choose the value of α, we compute the granular similarity based on the original arguments with the improved prompt on Llama for $\alpha \in \{0.3, 0.4, 0.5, 0.6, 0.7\}$. Table 7 shows that the highest granular similarity is obtained for $\alpha = 0.7$ in most cases, indicating that premises contribute more information to the argument.

text	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**
00	52.68	54.32	55.97	57.61	**59.26**
04	**63.4**	63.23	63.07	62.9	62.74
05	48.17	49.55	50.93	52.32	**53.7**
06	49.91	50.45	51	51.55	**52.09**
10	55.6	57.91	60.22	62.53	**64.85**
13	55.29	57.74	60.18	62.63	**65.07**
16	63.15	66.09	69.03	71.97	**74.91**
20	47.04	49.78	52.52	55.26	**58**
21	61.23	62.29	63.36	64.42	**65.48**

Table 7
Mean granular similarity for different values of α. Boldface correspond to the higher mean similarity, for each text.