Labwork Report : ABA framework & Binary argument classification

Abdurrahman SEN[1], Emilien Komlenovic[1], and Pierre Emmanuel Marrel[1]

[1]Université Claude Bernard Lyon 1

October 20, 2024

Abstract

This document is the report of our lab work on the subject of Argumentation Theory, in particular the automatic generation of an ABA+ framework and the process of binary classification on pairs of textual arguments. The results include : a web page where a user can input an ABA framework with preferences and compute the arguments and attacks ; a summary of results found in the literature regarding binary argument relation classification, and the results of our attempts at training a model for the same task.
Keywords: Argumentation Theory ; ABA framework ; RBAM

1 Introduction

This document is the report of our lab work on the subject of Argumentation Theory, in particular the automatic generation of an ABA framework and the process of binary classification on pairs of textual arguments.

1.1 Code repository

The repository used for this project is available on GitHub at the following address : https://github.com/ekomlenovic/APE

1.2 Setup

For the installation process, please refer to the README Markdown file available at the root of the repository.

2 ABA+

The code base has been implemented in python. We've made classes to implement an ABA framework :

- **Literal**: consists of a name and the possibility of being negative.
- **Rule**: consists of a head (literal) and a body (set of literals).
- **Argument**: consists of a name for representation (string), a claim (literal), and leaves (set of literals).
- **Contrary**: consists of a target and an attacker (literal).
- **Attacks**: consists of an attacker and a target (argument).
- **SetAttack**: consists of an attacker and a target (set of literals).

All classes can be hashed to make sets, so our ABA class uses this. Note that preferences are used like a dictionary. The keys are the literals, and their values are the literals that are less preferred. A certain function ensures the transitivity of the preference. For reverse and normal attacks, we start directly from the contraries to simplify the complexity of the function. All the computing

of an argument and attacks are done with the function "generate aba framework". To view our results you can visit our website here : https://ape-yz5e.onrender.com/.
The website might take a few minutes to load, because, with the free version of Render, the server shuts down when there is no activity on the website. The **website works** !

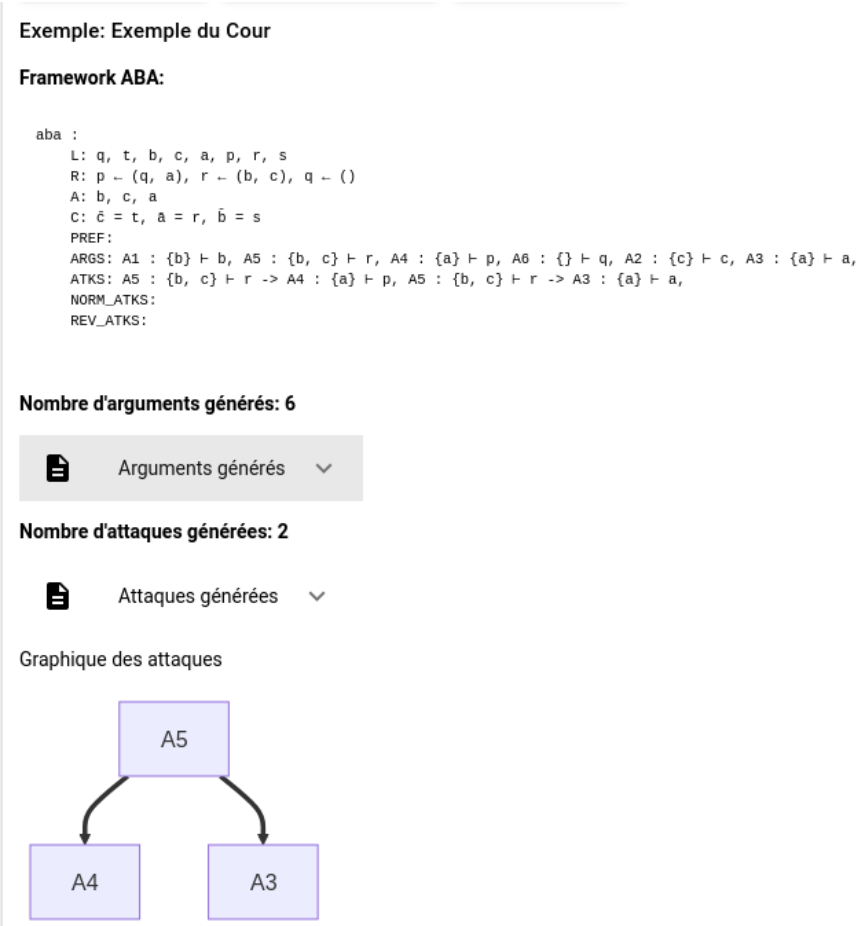

Figure 1: Website preview with an example taken form the course

All the computing is done as soon as an input is provided on the website.

3 Kialo data scrapping

3.1 Scrapping the data

In order to have data, we used the website Kialo(*Kialo* 2024). It is an online platform where users can create new debates and post arguments either for or against any other argument.

Using code examples by user 4mbroise found on GitHub (*4mbroise* 2024), we were able to collect 246 debates (and the arguments specified inside).

3.2 Data processing

After successfully scrapping the data, we encountered two issues with the data :

1. it includes debates in multiple languages, including : English, French, German, Italian, and Slovak
2. some debates include arguments with text similar to "See argument number X.X.X", which seem harder to contextualize (e.g. copy the argument text as is).

In order to solve the first, we used generated code to detect the language a debate was written in and separate them in different folders.

In order to solve the second, we dynamically remove such arguments from the dataset (pandas DataFrame) using a regex.

Metric	ACC	AUPRC	AUROC	F1 SCORE
Score	0.73	0.80	0.81	0.73

Table I: Evaluation results of our DistillBERT uncased model

4 Pairwise argument relation Binary classification

4.1 Existing approaches

We started our research of the existing approaches with the couple of papers already cited in Lecture 6, starting with Gorur et al. (Gorur *et al.*, 2024) who used RoBERTa and an architecture based on various LLMs and obtained an F1 score of 0.68 with RoBERTa and 0.75 with Llama70B-4bit (the best among evaluated LLMs).

Following this, the article from Ruiz-Dolz et al. (Ruiz-Dolz *et al.*, 2021) presented in the same lecture gives us another reference point of pairwise argument relation classification performance. In that paper, the authors trained various BERT-based models and obtained the highest F1 score of 0.70 with a RoBERTa model and an F1 score of 0.55 for a DistillBERT model.

Diving deeper into the article from Gorur et al. (Gorur *et al.*, 2024) allowed us to find more articles related to Argument Mining. Among those, the work of Bosc et al. (Bosc *et al.*, 2016) who focused on detecting arguments on Twitter and classifying pairs of tweets inside of an argument as either Attacks or Support. Their method obtained an F1 score of 0.16 for attacks and 0.20 for supports, which they explain is due to the specificity of the Twitter data and their model being tuned more towards the task of recognizing entailment.

The work of Cocarascu and Toni (Cocarascu and Toni, 2017) is directly related to our task and they achieved an accuracy of 0.89 and an F2 score of 0.89 with a LSTM model and allowing it to train the word embeddings.

Three years later, the authors (O. Cocarascu *et al.*, 2020) propose a set of baseline models trained and tested (for the task of relation prediction) on a variety of datasets used by researchers among the Argument Mining community. They find an average F1 score of 0.53 among all datasets.

4.2 Our approach

For this task, we were inspired by the Practical 4 and used the library SimpleTransformers (*SimpleTransformers* 2024).

We chose to use the Distilbert-base-uncased model (*distilbert-base-uncased* 2024), which only handles the English language and is very lightweight, yet still offers good performance. Our choice was mainly driven by hardware limitations. We had considered comparing our results with a model that supports multiple languages, thus avoiding the need for preprocessing the data, but we did not have enough time to pursue this.

The dataset used to perform the training was split into three (train, validation, test) following the ratios of 0.8 / 0.1 / 0.1 respectively.
The hyperparameters of the training include a context size of 256 tokens (i.e. words) and 10 epochs. Despite setting the number of epochs for training to 10, we had to stop it after 7 epochs in part because the model's results weren't improving after that, but also because of hardware limitations (training crashes).

The Table I presents the score metrics achieved with our trained model. When compared to the results found in the literature, our approach is definitely not the best (see (Cocarascu and Toni, 2017)) but pretty much around the average of most work found.

5 Indirect relation inference

The code processes a dataset of argument pairs with direct relations ("attack" or "support"). It aims to find new indirect relations by exploring the connections between arguments through a process of transitive reasoning.

For each argument pair, the code checks if there's an intermediate chain of relations between arguments (like A $\rightarrow$ B $\rightarrow$ C) and counts how many times an "attack" relation occurs. This is done recursively up to a certain depth (level of indirection). If indirect relations are found, they are

added as new argument pairs with labels, representing the cumulative effect of the intermediate relations.

	Distance 2	Distance 3	Distance 4	Distance 5	Distance 10	Distance 41
ACC	0.57	0.45	0.47	0.47	0.53	0.43

Table II: Evaluation results for different distances using our DistillBERT uncased model

The general trend across the distances shows that the model's performance declines as the distance increases. The accuracies fluctuate, but they largely remain around or below 50 percent, it suggests that the model is not effective at predicting the underlying relationships for indirect inference tasks.

In summary, while the model performs reasonably well for direct and one-hop relationships, it significantly struggles with longer indirect relationships.

6 Grades

We all agree to receive the same grade.
Abdurrahman SEN,
Emilien KOMLENOVIC,
Pierre-Emmanuel MARREL

References

4mbroise, (2024). [Accessed 14-10-2024]. **available at**: `https://github.com/4mbroise/ADC/blob/main/tool/kialo.ipynb`.

Bosc, Cabrio, and Villata (2016). "Tweeties Squabbling: Positive and Negative Results in Applying Argument Mining on Social Media", *Frontiers in Artificial Intelligence and Applications*, Vol. 287. DOI: `10.3233/978-1-61499-686-6-21`.

Cocarascu, Oana *et al.*, (2020). "A Dataset Independent Set of Baselines for Relation Prediction in Argument Mining", *CoRR*, Vol. abs/2003.04970. **available at**: `https://arxiv.org/abs/2003.04970`.

Cocarascu and Toni (2017). "Identifying attack and support argumentative relations using deep learning", *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 1374–1379.

distilbert-base-uncased, (2024). [Accessed 14-10-2024]. **available at**: `https://huggingface.co/distilbert/distilbert-base-uncased`.

Gorur, Rago, and Toni (2024). "Can Large Language Models perform Relation-based Argument Mining?", *ACL 2024*, **available at**: `https://arxiv.org/pdf/2402.11243`.

Kialo, (2024). [Accessed 14-10-2024]. **available at**: `https://www.kialo.com/explore/featured`.

Ruiz-Dolz *et al.*, (2021). "Transformer-Based Models for Automatic Identification of Argument Relations: A Cross-Domain Evaluation", *IEEE Intelligent Systems*, Vol. 36 No. 6, pp. 62–70. DOI: `10.1109/MIS.2021.3073993`.

SimpleTransformers, (2024). **available at**: `https://simpletransformers.ai/`.