July 2020

Towards an Argument Mining Pipeline Transforming Texts to Argument Graphs

Mirko LENZ [1], Premtim SAHITAJ, Sean KALLENBERG, Christopher COORS, Lorik DUMANI, Ralf SCHENKEL, and Ralph BERGMANN
Trier University, Trier, Germany

Abstract. This paper tackles the automated extraction of components of argumentative information and their relations from natural language text. Moreover, we address a current lack of systems to provide a complete argumentative structure from arbitrary natural language text for general usage. We present an argument mining pipeline as a universally applicable approach for transforming German and English language texts to graph-based argument representations. We also introduce new methods for evaluating the performance based on existing benchmark argument structures. Our results show that the generated argument graphs can be beneficial to detect new connections between different statements of an argumentative text.

Keywords. computational argumentation, argument mining, argument graph construction, argument graph metrics

1. Introduction

Argumentation plays an integral role in many aspects of daily human interaction. People use arguments to form opinions, discuss ideas or change the views of others. Many resources dealing with argumentation are available, but the content is mostly unstructured. Due to the current capabilities of modern hardware, Computational Argumentation (CA) is a field of increasing interest. While previous work [1,2] has focused rather on individual tasks such as claim detection [3], this paper targets the automated extraction of argumentative components and their relations from natural language text. We address a gap in the argument mining field where end-to-end pipelines that generate complex argument structures for CA are not prevalent. We present such a pipeline that provides a universally applicable approach for transforming German and English language texts to graph-based representations [4] in the popular AIF format [5]. We also introduce new methods for evaluating results based on benchmark data and present a new argument graph corpus.

2. Foundations and Related Work

Argumentation, in a formal way, is described as a set of arguments in texts. An argument is constructed by at least two *Argumentative Discourse Units* (ADUs) which represent

[1]Corresponding author. E-mail: info@mirko-lenz.de

different components of argumentation, e.g. claims and premises. Additionally, we can represent the stance between two ADUs as a supporting or attacking directed relation. A *major claim* is defined as the claim that describes the key concept in an argumentative text [2]. An *argument graph* describes a structured representation of argumentative text [6]. We use a variant of the well-known Argument Interchange Format (AIF) [5], extended to support the explicit annotation of a major claim M [7]. Claims, premises, and the major claim are represented as *information nodes* (I-nodes) I while relations between them are represented by *scheme nodes* (S-nodes) S. We define an argument graph G as triple $G = (V, E, M)$ with a set of nodes $V = I \cup S$ and a set of edges $E \subseteq V \times V$.

We aim at addressing the research gap of a general-use end-to-end pipeline for the German and English languages by following and extending the approaches of related work in the field. Cabrio and Villata [8] define the central stages of an argument mining framework to be argument extraction and relation prediction. Stab and Gurevych [2] present an approach for extracting arguments by identifying ADUs with further classification into major claim, claims, and premises by considering structural, lexical, syntactical, and contextual features [3]. The segmentation of natural language text into ADUs is simplified by considering textual boundaries on the sentence level [9]. Many researchers formulate relation prediction as a binary classification problem to distinguish between support and attack [10]. The argumentative information is then used to construct an argument graph from the extracted ADUs [8]. To the best of our knowledge, only Stab and Gurevych [2] addressed a method to link ADUs within the same paragraph in an argumentative text. Nguyen and Litman [11] developed a specialized end-to-end argument mining system that includes the identification of relevant ADUs, the classification of components as well as the prediction of their relations.

To assist argument mining techniques, a diverse selection of corpora exists. Stab et al. [12] and Eger et al. [13] provide a corpus with 402 annotated persuasive essays—in the following called PE. It consists of $11,078$ nodes and $10,676$ edges. Another corpus has been developed by the ReCAP project [14], composed of 100 argument graphs dealing with educational issues in Germany. It consists of $4,814$ nodes and $4,838$ edges.

3. Argument Mining Pipeline

The pipeline introduced by Nguyen and Litman [11] is used as the basis of our proposed architecture and extended by a novel graph construction process. Our pipeline is designed in a modular way where each step describes an individual and interchangeable module.

Argument Extraction As a first step, the input text is segmented into sentences [2,14]. Then, multiple types of features are extracted, derived from Stab and Gurevych [12] as well as Lippi et al. [15], depicted in our GitHub project.[2] The basis of the entire approach is the correct identification of ADUs. Based on these features, the sentences are classified into argumentative and non-argumentative units. The ADUs are then further categorized into claims and premises using a separate classifier.

[2] `https://github.com/ReCAP-UTR/Argument-Graph-Mining`, licensed under Apache 2.0.

Relationship Type Classification To construct an argument graph from a natural language text, it is necessary to consider the task of textual entailment. Here, we assign the relation type between the identified ADUs [16]. We consider only the inference from premises to claims. Due to the complexity of considering a multi-class stance problem and the lack of training data of more sophisticated argument schemes (e.g., Walton et al. [17]), we train a model to only classify attacking and supporting relations. GloVe embeddings are used as the only feature for this task to focus on semantic information. Based on the model's metadata, we detect indifferent results (i.e., having a classification probability below a configurable threshold). In this case, the type support is used.

Major Claim Detection A very crucial step in the graph generation is the location of the major claim. Neither pretrained models nor sufficient training data are available, as each text usually has only one major claim, regardless of its length, making machine learning-based approaches infeasible. The classifier by Stab et al. [2] cannot be applied as it condenses all classification steps into a single model, which does not fit our proposed pipeline. Thus, we examine the following heuristics:

FIRST: The first claim based on the text position is chosen as the major claim. This is done because the main argument is often referred to in the introduction or headline (e.g., Dumani et al. [14]). CENTROID: When treating the major claim as the core proposition of the text, we can assume that it should be very similar to all ADUs. Thus, we can compute the centroid of all embeddings to estimate the core message. The major claim is then the ADU with the highest cosine similarity to the centroid. PAIRWISE: Pairwise cosine similarity of all embeddings of the ADUs is computed. The major claim is defined as having the highest average similarity to all other ADUs. The rational for this technique is similar to CENTROID. PROBABILITY: Again, a cross product of all ADUs is computed. Based on the relationship classification (see above), the major claim is defined as having the highest average classification probability except for neutral results, (i.e., we select the ADU where the model shows the highest certainty in all of its predicted relations).

Graph Construction Utilizing the acquired information, we can now construct the graph. To the best of our knowledge, there is no automatic procedure that links ADUs to complex graphs. We propose three algorithms to address this task. In all cases, ADUs are used as I-nodes and the S-nodes between them are derived from the relationship type classification. As a simplification, the major claim is set as the root.

FLAT TREE: Our baseline approach connects all ADUs as I-nodes to the major claim using the predicted S-nodes, resulting in a two-layer graph. While not suitable for complex texts, it may still provide sufficient results for smaller ones. ADU POSITION: This technique makes use of typical argument compositions. We assume that premises belonging to a claim are contained in the same paragraph and thus positioned in close proximity of the claim in the original text [2]. In the first step, all claim I-nodes are connected to the major claim using the respective S-nodes. Then, each premise I-node is connected to the nearest claim via an S-node. If no claim is detected, all premise I-nodes are connected directly to the major claim via S-nodes. The resulting graph consists of at least two and at most three layers. PAIRWISE COMPARISON: This method leverages the class probabilities of the relationship type classification. Its idea is to draw an edge between ADUs whose relation probability is above a certain threshold. First of all, tuples of ADUs (a,b) are computed such that b has the highest relation probability among all possible connections of a. If multiple ADUs reach the same maximal value, the first one is chosen. Then,

a configurable lower bound (in our case 0.98) below this maximal probability is defined. Each ADU related to the major claim with a score above the lower bound is connected as an I-node via a corresponding S-node. If the major claim has no connections after this step, the ADU that first occurs in the text is used as an I-node and connected to the major claim. Then, the remaining ADUs are connected iteratively (via S-nodes) to the I-node where their score is above the lower bound. If there remain ADUs not used after a certain amount of repetitions, they are connected to the major claim using a support S-node.

4. Experimental Evaluation

In this section we evaluate our end-to-end approach by assessing the resulting argument graph structures. Moreover, we compare the correspondence of our automatically generated graph to a given benchmark graph.

Hypotheses The following hypotheses, covering all aspects of the pipeline, will be tested in our evaluation: **(H1)** Using sentences as an argumentative unit yields a robust approximation of the manual segmentation. **(H2)** Selecting the major claim using FIRST will give the best results as it reflects common argumentation patterns. **(H3)** Using a threshold for the relationship type classification (i.e., a value above 0.5) will perform best as supporting arguments occur more often than attacking ones. **(H4)** Using ADU POSITION to construct graphs will result in the best approximation of the benchmark data due to the claim-premise information. **(H5)** Providing the pipeline with predefined ADUs will result in graphs that better reflect the human annotation than end-to-end graphs.

Experimental Setup and Datasets The implementation has been done in Python and is available on GitHub. Three datasets are used for the evaluation: ReCAP, PE (see Section 2) and a new one created for our tasks. The ReCAP corpus contains fragments such as headlines and metadata that were removed manually from the input files. We are using two versions of the PE dataset. PE_{17} is based on Stab et al. [12]. The length of the ADUs differs greatly and is not in line with our sentence-based segmentation. PE_{18} is based on Eger et al. [13] and was transformed by us from word- to sentence-based labels to conform to our segmentation approach. A major difference is that PE_{17} has information about relations between ADUs (i.e., available as argument graphs), while PE_{18} only provides the ADUs. We also explored the open discourse platform `kialo.com` due to the availability of much larger argument graphs. We extracted the 589 debates in the popular collection (as of Jan. 2020), consisting of $190,269$ I-nodes, $189,680$ S-nodes and $379,360$ edges. The data is available in English and German (translated via `deepl.com`) on request from the authors.

Classification Models For ADU and claim - premise classification we chose an ensemble stacking method build from a layer of a logistic regression, random forest and adaptive boosted decision tree [18] as they were shown to perform well for those specific tasks [19]. The classifiers' first layer adds their predictions as feature to the input features and passes them on to the final estimator which provides the output prediction. For the output layer we chose extreme gradient boosted random forest [20]. The ADU model was trained using the PE_{18} and ReCAP datasets in their respective native languages (i.e., German for ReCAP and English for PE_{18}) to mitigate any translation errors. The claim-premise classifier was trained using PE_{18} for both languages as it is the only one that

Table 1. Results of the ADU and claim-premise classification. $A :=$ Accuracy, $P :=$ Precision, $R :=$ Recall

(a) ADU model.

Language	A	P	R	F_1
English (PE_{18})	0.80	0.80	1.0	0.89
German (ReCAP)	0.54	0.52	0.66	0.58

(b) Claim-premise model.

Language	A	P	R	F_1
English (PE_{18})	0.52	0.52	0.68	0.59
German (PE_{18})	0.76	0.73	0.13	0.22

differentiates between claims and premises while also using sentences as units. To eliminate biases, a 90/10 train/test split has been performed before training. The models were trained through a 5-fold stratified cross-validation on the training set and tuned through a random search. The reported values are results from a single evaluation on the test set.

We observed that the ADU classification reached highly varying results between the two datasets. Probably the limited quantity of training data in the ReCAP dataset is the main reason for the variation. On the more than four times larger essay data we obtained an accuracy score of 0.80 which yields a strong indication of the model's generalization ability. The claim-premise classification unfortunately did not meet expectations on neither the persuasive essays nor on the ReCAP data. We explain the difference in predictive power on both datasets due to the fact that the structure of the ReCAP dataset is too dissimilar to the PE_{18} dataset on which the models are trained on. In Table 1 we report accuracy A, precision P, recall R and F_1 values for the used classification models.

The training of the relationship type model was done with the Kialo dataset due to the large number of available relations. The triples were split into 70% training and 30% testing data. Among state-of-the-art classifiers, extreme gradient boosting achieved the highest accuracy for both languages with 0.678 and 0.668 for the English and German language, respectively. Logistic regression performed very similar (0.672 and 0.664) while being computationally simpler, leading us to choose the latter.

Argument Graph Metrics To assess the quality of the entire pipeline as well as its individual steps, multiple metrics are needed. We are not aware of existing measures that enable the verification of our hypotheses and thus introduce a novel approach. For each element in the benchmark graph (i.e., I-nodes, S-nodes, major claim and edges), the corresponding item in the generated graph is determined to compute an agreement.

To compare the ADU segmentation, we need a mapping between the I-nodes of the benchmark graph G_b and the generated graph G_g. It is based on the Levenshtein distance [21] $\mathrm{dist}(u_b, v_g)$ between the benchmark I-node u_b and the generated I-node v_g and the derived similarity $\mathrm{sim}(u_b, v_g) = 1 - (\mathrm{dist}(u_b, v_v)/\max\{|u_b|, |v_g|\})$. The mapping $m\colon u_b \mapsto v_g$ assigns each I-node of the benchmark graph an I-node of the generated graph s.t. their similarity is higher than any other combination of I-nodes. In case that two generated nodes have the same similarity to the benchmark node, we pick the first one. If the ADU segmentation between the benchmark and generated graph differs, the benchmark node is mapped to the generated node having the highest similarity while ignoring the other nodes. The *I-nodes agreement* $\mathscr{I}$ is defined by the weighted arithmetic mean of the similarity between the benchmark I-nodes and their respective mappings. The *major claim agreement* $\mathscr{M}$ is specified as a binary metric that is 1 iff the major claims are mapped or there is none defined in the benchmark and 0 otherwise.

For the evaluation of S-nodes, we need to consider the surrounding I-nodes, because S-nodes do not contain textual content that could be used for similarity assessments. We

compute all combinations of connections of the benchmark S-node $\text{in}(u_b) \times \text{out}(u_b)$ and determine individual tuples based on their respective mappings as $(m(\text{in}), m(\text{out}))$. Using this information, it is possible to compare the benchmark S-node with the information provided by the relationship type classification. The *S-node agreement* $\mathscr{S}$ is then defined as the number of correctly classified relationships divided by the total number of tuples.

Lastly, edges need to be considered as well. As they do not contain textual information, we use the triple (x, y, z) where x and z represent I-nodes and y an S-node. Thus, we consider two edges at a time. The two edges in the benchmark graph are mapped to their counterparts in the generated graph if they connect the same I-nodes (as determined by the mapping m). The direction of the edges is not relevant. The S-node y is ignored deliberately to mitigate potential errors during earlier tasks. The *edges agreement* $\mathscr{E}$ is determined by dividing the number of mapped edges by the total number of edges.

5. Results and Discussion

We will now evaluate the pipeline using the test splits of the German ReCAP corpus and the English PE corpora. Exemplary cases can be found in the extended version [22].

German ReCAP Corpus The test set for the ReCAP corpus contains ten texts with benchmark graphs. We get an *I-node* agreement $\mathscr{I} = 0.461$ for all possible combinations of parameters. In most cases, there were fewer, but larger ADUs in the generated graph compared to the benchmark. This stands in contrast to the fact that the average ADU length in the ReCAP corpus is 1.1, indicating mismatches in the definition of a sentence, for example due to punctuation. It also contradicts **H1**. Table 2a shows the results of the three *major claim* detection approaches. They are very similar, differing only in one case (as we have exactly one major claim per text). The two best methods CENTROID and PAIRWISE predicted exactly the same major claims. As FIRST performed worst here, **H2** might be rejected. All thresholds for the *relationship type* classification are depicted in Table 2b. The best result can be obtained using 1.0 (i.e., the classifier always predicts support), which means that almost all of the relations in the benchmarks are of the type support. With such a skewed distribution, this corpus may not be suitable to assess **H3**, thus we will postpone it to PE. When comparing end-to-end with preset ADUs, we observe that the latter one delivers slightly worse performance with all thresholds above 0.6. This could be caused by the smaller preset ADUs which provide less contextual information for the classifier. This stands in slight contrast to **H5**. Lastly, Table 2c shows the three *graph construction* methods. As the scores depend on the major claim method, we used the best approach (i.e., CENTROID/PAIRWISE) for the end-to-end graph. The algorithm FLAT TREE delivered the best results across the board, contradicting **H4**. As expected, the scores themselves are very low, especially for the end-to-end graph, making manual examination of individual edges necessary. When comparing the end-to-end graph with the one using preset ADUs, we notice a major increase in the agreement score. Using the best method, almost half of the edges were connected correctly, providing support for **H5**. This is in large part caused by using the correct major claim as the root node.

English PE Corpus For the following evaluation, the test split (see Section 4) of the PE corpus is used, consisting of 40 cases. The results of PE_{17} are very similar to the findings of the ReCAP corpus. The I-node agreement $\mathscr{I} = 0.622$ is higher than for the

Table 2. Aggregated results of the evaluation using the ReCAP corpus.

(a) Major claim methods.

Method	$\mathscr{M}$
CENTROID	**.200**
FIRST	.100
PAIRWISE	**.200**
PROBABILITY	.100

(b) Relationship type thresholds.

Threshold	$\mathscr{S}_{\text{e2e}}$	$\mathscr{S}_{\text{preset}}$
0.5	.460	.514
⋮	⋮	⋮
0.9	.927	.898
1.0	**.937**	**.902**

(c) Graph construction methods (CENTROID major claim for e2e).

Method	$\mathscr{E}_{\text{e2e}}$	$\mathscr{E}_{\text{preset}}$
ADU POSITION	.064	.166
FLAT TREE	**.095**	**.449**
PAIRWISE COMP.	.054	.296

ReCAP graphs, providing support for **H1**. CENTROID and PAIRWISE performed best for identifying the major claim ($\mathscr{M} = 0.1$), contradicting **H2**. A threshold of 0.9 for the relationship type classification yields the highest agreements ($\mathscr{S}_{\text{e2e}} = 0.936$ and $\mathscr{S}_{\text{preset}} = 0.912$). Again, the S-node distribution is skewed, but as two different corpora show the same results, we can accept **H3** for certain corpora. The best edge agreement scores can be obtained using ADU POSITION for the end-to-end graph ($\mathscr{E}_{\text{e2e}} = 0.130$) and FLAT TREE for the graph with preset ADUs ($\mathscr{E}_{\text{preset}} = 0.274$). All graph construction methods show a low agreement, thus **H4** needs to be rejected. The use of preset ADUs provides a benefit in the edge agreement with only a small decrease in the S-node agreement, leading to the final acceptance of **H5**. Overall, the findings show the robustness of the proposed approach for varying input data. The PE_{18} dataset provides another perspective on the pipeline by using sentence-based segmentation. The I-node agreement $\mathscr{I} = 0.799$ shows a decent approximation of the segmentation, leading to the partial acceptance of **H1** for certain corpora (e.g., essays). The *major claim* agreement $\mathscr{M}$ is 0.125 for CENTROID and PAIRWISE, 0.175 for PROBABILITY and 0.250 for FIRST. As FIRST was only best in this specific corpus and the values are low overall, we have to reject **H2**.

6. Conclusion and Future Work

In this work, we investigated new methods towards the automated mining of argument graphs from natural language texts for both English and German. The pipeline successfully extends previous approaches [11] by generating even complex graphs as end product. Our results show that there are great differences in the resulting graphs based on the type of input data. For very homogeneous corpora such as PE, the agreement is very high, but in heterogeneous datasets such as ReCAP, the methods performed rather poor. When looking beyond the goal to approximate a human annotation as much as possible, the generated graphs might be very beneficial to detect new connections between single statements of an argumentative text. Using multiple methods to construct different representations from a single text might also help in educating professional annotators by discussing the strengths and weaknesses of individual cases.

In future work we plan to provide a more flexible approach for segmenting a text into potential ADUs. A limitation of the current evaluation procedure lies in the edge agreement, which could be tackled by providing multiple benchmark graphs to account for uncertainty. As the ReCAP corpus makes use of detailed argumentation schemes [17], the pipeline should be extended make use of them. Finally, we will investigate the potential use of argument graphs for the task of measuring argument quality [23] in unstructured texts through the use of argument mining.

Acknowledgments This work has been funded by the Deutsche Forschungsgemeinschaft (DFG) within the project *ReCAP*, Grant Number 375342983 (2018–2020), as part of the Priority Program "Robust Argumentation Machines (RATIO)" (SPP-1999). We would also like to thank *DeepL* for providing free access to their translation API.

References

[1] Levy R, Bogin B, Gretz S, Aharonov R, Slonim N. Towards an argumentative content search engine using weak supervision. In: COLING; 2018. p. 2066–2081.
[2] Stab C, Gurevych I. Identifying Argumentative Discourse Structures in Persuasive Essays. In: EMNLP; 2014. p. 46–56.
[3] Lippi M, Torroni P. Argument Mining from Speech: Detecting Claims in Political Debates. In: AAAI; 2016. p. 2979–2985.
[4] Craven R, Toni F. Argument Graphs and Assumption-Based Argumentation. Artificial Intelligence. 2016;233:1–59.
[5] Chesñevar C, McGinnis J, Modgil S, Rahwan I, Reed C, Simari G, et al. Towards an Argument Interchange Format. Knowl Eng Rev. 2006;21(4):293–316.
[6] Stede M, Afantenos SD, Peldszus A, Asher N, Perret J. Parallel Discourse Annotations on a Corpus of Short Texts. In: LREC; 2016. .
[7] Lenz M, Ollinger S, Sahitaj P, Bergmann R. Semantic Textual Similarity Measures for Case-Based Retrieval of Argument Graphs. In: ICCBR. vol. 11680 of Lecture Notes in Computer Science; 2019. p. 219–234.
[8] Cabrio E, Villata S. Five Years of Argument Mining: a Data-driven Analysis. In: IJCAI; 2018. p. 5427–5433.
[9] Stab C, Miller T, Schiller B, Rai P, Gurevych I. Cross-topic Argument Mining from Heterogeneous Sources. In: EMNLP; 2018. p. 3664–3674.
[10] Stab C, Gurevych I. Parsing Argumentation Structures in Persuasive Essays. Computational Linguistics. 2017;43(3):619–659.
[11] Nguyen HV, Litman DJ. Argument Mining for Improving the Automated Scoring of Persuasive Essays. In: AAAI; 2018. p. 5892–5899.
[12] Stab C, Gurevych I. Parsing Argumentation Structures in Persuasive Essays. Computational Linguistics. 2017 Sep;43(3):619–659.
[13] Eger S, Daxenberger J, Stab C, Gurevych I. Cross-lingual Argumentation Mining: Machine Translation (and a bit of Projection) is All You Need! In: COLING; 2018. p. 831–844.
[14] Dumani L, Biertz M, Witry A, Ludwig AK, Lenz M, Ollinger S, et al.. The ReCAP Corpus: A Corpus of Complex Argument Graphs on German Education Politics; 2020.
[15] Lippi M, Torroni P. Context-Independent Claim Detection for Argument Mining. In: IC-AI. IJCAI'15; 2015. p. 185–191.
[16] Cabrio E, Villata S. Combining Textual Entailment and Argumentation Theory for Supporting Online Debates Interactions. In: ACL; 2012. p. 208–212.
[17] Walton D, Reed C, Macagno F. Argumentation Schemes; 2008.
[18] Schapire RE. A Brief Introduction to Boosting. In: IJCAJ. IJCAI'99; 1999. p. 1401–1406.
[19] Aker A, Sliwa A, Ma Y, Lui R, Borad N, Ziyaei S, et al. What works and what does not: Classifier and feature analysis for argument mining. In: ArgMining@EMNLP; 2017. p. 91–96.
[20] Chen T, Guestrin C. XGBoost: A Scalable Tree Boosting System. CoRR. 2016;abs/1603.02754.
[21] Levenshtein VI. Binary Codes Capable of Correcting Deletions, Insertions and Reversals. Soviet Physics Doklady. 1966 Feb;10:707.
[22] Lenz M, Sahitaj P, Kallenberg S, Coors C, Dumani L, Schenkel R, et al. Towards an Argument Mining Pipeline Transforming Texts to Argument Graphs. arXiv: 2006 04562. 2020;.
[23] Wachsmuth H, Naderi N, Hou Y, Bilu Y, Prabhakaran V, Thijm TA, et al. Computational Argumentation Quality Assessment in Natural Language. In: EACL; 2017. p. 176–187.