Computational Models of Argument
C. Reed et al. (Eds.)

doi:10.3233/FAIA240308

Parsing Graphical Summaries from Argumentative Dialogues

Jonathan CLAYTON [a,1], Marco DAMONTE [b] and Robert GAIZAUSKAS [a]
[a] *University of Sheffield*
[b] *Amazon*

Abstract. In this paper, we introduce a novel Argument Mining task based on the existing task of Argument Structure Parsing (ASP). Our new task, which we call *ASG Parsing*, is the task of generating Argument Summary Graphs (ASGs) from dialogical argumentative text. We release a dataset containing ASGs, a type of graphical summary for argumentative dialogues, in which the nodes are summaries of statements and the edges are the argumentative relations between them (*support* or *attack*). We approach the problem with two different LLM-based solutions: (a) a pipeline system involving two models separately fine-tuned for summarisation and stance detection; and (b) an end-to-end system based on the TANL (Translation between Augmented Natural Languages) framework [1]. We show that the TANL approach outperforms the pipeline approach across the board. We also show that, for all systems, performance degrades as the depth of the graphs increases.

Keywords. Argument Mining, Summarisation, Stance Detection, TANL

1. Introduction

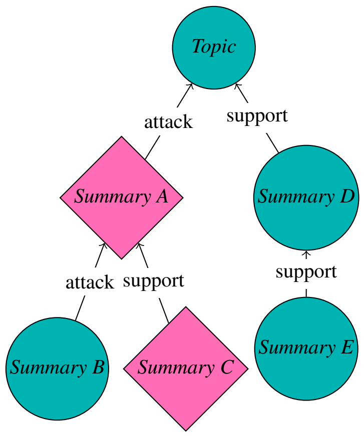
Argumentative dialogue is a topic which is currently of interest due to the increasing volume of online debates and the degree to which online argument is now playing a role within public discourse and shaping politics [2].

Much work in Argument Mining has focused on Argument Structure Parsing (ASP), the task of identifying argumentative propositions and their relations in an argumentative text (see Section 2.2). However, when it comes to analysing an often lengthy, repetitive and logically poorly structured multi-party dialogical text such as an online newspaper comment section or discussion forum, it is clear that summarisation is also important (as discussed in Section 2.1).

In this paper we propose a novel task, which we refer to as Argument Summary Graph (ASG) Parsing, a task combining summarisation and ASP. Given an argumentative text, the task is to produce a graphical summary of the main arguments present in the text.

An example ASG is shown in Figure 1. Unlike in a standard ASP task, where the output must contain every premise and conclusion within the argument, an ASG contains only key points and the support/attack relations which hold between them. In this paper we examine two research questions that we have addressed in the work we have carried out on ASG parsing up to this point:

[1]Corresponding Author: Jonathan Clayton, jclayton2@sheffield.ac.uk



Topic: Housewives should be paid for their work.

Comment A: By paying housewives for their work, you create negative stereotypes about families and women by commodifying the role of home-keeper. Paying housewives for their work re-enforces the...

Comment B: Paying housewives a wage would improve not reduce social mobility. Many women would still choose to go to university and the vast majority who do will still want to work. Paying housewives...

Comment C: The result is that women are discouraged from seeking to fulfil their own dreams by creating their own careers as they are more firmly chained to their traditional role. This is damaging to societal views...

Comment D: It is estimated that the value of a housemaker's services would be equivalent to approximately £30,000 per year. In the same way that any product or service is created...

...

Summary A: Paying housewives reduces social mobility
Summary B: It will improve women's financial freedom
Summary C: Reinforces social expectations for women
Summary D: Housewives are entitled to pay
...

Figure 1. An example Argument Summary Graph that could be generated from an online debate. Summaries corresponding to each node are shown below the tree for clarity. Summaries in ● support the top-level comment (Topic); summaries in ◆ argue against. Example from `idebate.net`[2].

1. Are pipelines of summarisation and argument parsing better suited than end-to-end solutions like Kawarada et al. [3] that are the state of the art in ASP?
2. How much does performance degrade when increasing the depth of the generated output trees?

The main contributions of this work are as follows:

- We propose ASG Parsing, a task combining summarisation and Argument Structure Parsing.
- We release a dataset for the ASG task, which is derived from 590 debates available on the *Debatabase* website [3].
- We compare the effectiveness of a multi-stage LLM pipelines to a single end-to-end model using the TANL (Translation between Augmented Natural Languages) framework [1]. We show that the TANL models substantially outperform the pipeline approach across different LLM models and metrics.
- We compare two different tree depths (depth-1 and depth-2). We show that performance for all models degrades with even this slight increase in tree depth, highlighting a limitation of the approaches expored here.

2. Background

Work on extracting graphical structures from argumentative text using Natural Language Processing techniques is a growing field. In Section 2.1, we motivate the Argument Summary Graph parsing task, in Section 2.2, we look at previous approaches to ASP, and in Section 2.3, we look at existing corpora relevant to our proposed task.

[2]https://idebate.net/this-house-believes-that-housewives-should-be-paid-for-their-work~b1087
[3]idebate.net

2.1. Task Motivation

Online comment threads have several attributes which make them less accessible to a casual reader who wants to quickly get an overview of the topic under discussion. These include the fact that often they (a) are very long, (b) contain redundant comments, and (c) are not necessarily organised in a logical order.

We propose a logically organised, graphical representation for online comment threads. Specifically, we are interested in *argumentative* comment threads (that is, comment threads which contain a debate) and therefore propose a type of graph which can display the argumentative relations between comments (in our case "support" or "attack"). These structures are fundamentally the same as the argument maps found on `kialo.com` [4], which themselves bear similarities to the argument mapping techniques developed in works such as Freeman [5] and Budzynska et al. [6]. Throughout the paper, we will refer to these structures as Argument Summary Graphs or ASGs.

ASGs are designed to allow readers of the dialogue to more easily and quickly obtain a broad overview of the most important issues under discussion without the need to read a large number of online comments. This may be useful for a general audience interested in knowing the opinions of the readership of a certain online forum or newspaper on a particular controversial topic. Additionally, this might benefit more specialised audiences who have an interest in the automatic analysis of public opinion such as journalists and social scientists. In order to have an intuition for how such graphs might be helpful, the reader can consult the manually-created interactive argument graphs on `kialo.com` [4], which illustrate the type of interface which could be used to interact with an automatically-generated ASG.

2.2. Argument Structure Parsing

Although our task differs from Argument Structure Parsing, it shares some important similarities, and therefore it is worth examining the different sub-tasks of ASP and the approaches that have been taken to them. Argument Structure Parsing includes four key sub-tasks: (i) Argument Component Segmentation (ACS), (ii) AC type classification (ACTC), (iii) Link Identification (LI), and (iv) Link Type Classification (LTC) (we take the terminology from Kuribayashi et al. [7]).

ACS is the task of extracting argumentative spans from text, with those spans typically being argumentative premises or conclusions. ACTC is the task of classifying them, typically into categories like "premise" or "conclusion". LI involves identifying which of the extracted spans have rhetorical links which hold between them. Finally, LTC involves classifying what type of link exists, for example, "support" or "attack".

Due to the complexity of this task, many different approaches have been taken. The earlier approaches involve classifier models such as Support Vector Machines, Random Forests and Maximum Entropy classifiers [8, 9], and Integer Linear Programming [8, 9]. A great deal of work has also applied neural models to this task, including BiLSTMs [10], pointer networks [11] and a wide variety of different types of pretrained transformer network (e.g. [12, 13, 14]).

Recently, Kawarada et al. [3] have obtained state-of-the art results on several ASP corpora using the TANL or Translation between Augmented Natural Languages framework[1]. This is a general framework for structured prediction tasks in NLP. In contrast to a standard classification model, which is trained to produce one or more labels for a span of text, in TANL the LLM is trained to produce annotated text which contains the labels within it. In the case of Kawarada et al. [3], the labels in the output text are generated in this format:

> [Living in a new country requires a great amount of flexibility and adaptability in one' s character | premise | support = students who study outside their countries can get a lot of experience living in a foreign country]

In this text we see three sections separated by pipe symbols: firstly, the text span being annotated; secondly, the argumentative component class of this span; and then finally, the "parent" text span in the argument graph (i.e. the proposition that the span in question either supports or attacks) and the relation to this parent. We apply a somewhat similar annotation scheme in our work, shown and described in Section 5.

2.3. Existing Corpora

Important work in developing monological argument corpora was done by Stab and Gurevych [15] and Peldszus and Stede [16], both of whom report corpora containing short essays or essay-like texts, annotated with argumentative text spans and the links between these spans. The labels of these spans vary between corpora, but typically they include a "Premise" label and a "Claim" label, and different argumentative relations which can hold between Claims and Premises, for example, a premise may "support" a claim.

Additional work has attempted to look at multi-speaker or multi-user corpora. The Internet Argument Corpus [17] contains argumentative online forum posts, each annotated with a label stating whether the post is "pro" or "con" a given claim, but does not contain textual summaries for individual comments. The corpora contained in AIFdb [18], for example Hautli-Janisz et al. [19], contain a number of dialogic texts, such as radio debates, with rich annotations including "attack" or "support" relations between speakers. VivesDebate [20] is a trilingual corpus fulfilling a similar role. However, these corpora again do not contain summaries. Another prominent corpus for the summarisation of debate is DebateSum [21]; however, in contrast to the above-mentioned corpora, this dataset is purely text-summarisation based and does not contain any argumentative relation information.

Outside Argument Mining narrowly defined, one corpus which is relevant to our work is the SENSEI annotated corpus [22]. This corpus contains a number of online newspaper comment sections, many of which contain argument, together with summaries of comment clusters. However, no argumentative relation annotations have yet been produced for this corpus.

All these corpora differ from the one which we generate for this work: while there are corpora containing either summaries of arguments or argumentative relations, there is (at least to our knowledge) no available corpus containing both.

3. ASG Parsing

The proposed task, ASG parsing, is as follows: given a comment thread, containing an argumentative discussion on topic *t* and containing `n` comments, generate an ASG with `n+1` nodes, where the root is the topic *t* and the remaining `n` nodes are summaries of each comment, as shown in the example of Figure 1.

Our argument graphing scheme is very simple, inspired by a simplified version of Freeman's argument graphing formalism [5] and the graphs on the Kialo debate visualisation website [4]. The ASG is tree structured, with the root of the tree being the topic t. Each of the non-root nodes must contain a summary of the corresponding comment. The argumentative relations between the nodes are labelled as "support" or "attack" - this represents the stance of the child node towards its parent node.

The motivation for the use of a tree structure instead of a less constrained graph type is due to an assumption about the argumentative discussion: disagreements on a particular topic branch downwards to disagreements about a sub-topic that the parent argument depends on.

Although other authors have addressed the task of generating non-tree argument graphs (e.g. [12]), this goes beyond the scope of our task as it would complicate the syntax of the graphs we are requiring the LLM to generate. It would also conflict with our aim of producing a simple, human-readable graphical summary of an argument.

It is also important to note that discussion on online comment threads "in the wild" often do not revolve around a single controversial topic, particularly in contexts such as discussion of a news article, in which there could be several important points of contention. Therefore our work presupposes an initial topic clustering step (not addressed here) in which comments are organised into one or more topically coherent clusters, for each of which an ASG is then generated. Previous work has been carried out on this task, e.g. on the SENSEI dataset [23].

We have also made the simplification that every comment in the thread must either support or attack the main topic t. In real discussions, it is possible for a single comment to contain a mix of propositions supporting and attacking t, and this is an issue which may require modification of our chosen formalism in future work.

4. Evaluation

We aim to evaluate three aspects of the correctness of the output summary graphs:

1. The quality of the textual summaries at each node
2. The correctness of the graph structure
3. The correctness of the support/attack labels

To evaluate each of these three aspects, we use three different metrics: ROUGE, Graph Edit Distance, and F1 measure on the predicted node stance (Node-Stance-F1).

ROUGE is the de-facto standard for measuring the quality of summaries. The specific variation that we use is Rouge-2 (bigram Rouge) [24].

Graph Edit Distance [25] is a metric similar to the Levenshtein distance [26] but for graphs instead of strings. The graph edit distance between two graphs g_1 and g_2, $GED(g_1, g_2)$ can be defined as

$$GED(g_1,g_2) = \min_{(e_1,\ldots,e_k)\in\mathcal{P}(g_1,g_2)} \sum_{i=1}^{k} c(e_i)$$

where $\mathcal{P}(g_1,g_2)$ denotes the set of edit paths transforming g_1 into g_2, $(e_1,\ldots,e_k)$ is a sequence of graph edit operations comprising an edit path, where each e_i is either an edge insertion or edge deletion operation and $c(e)$ is the cost of each graph edit operation e. Nodes are aligned based on the predicted comment labels (e.g. "Comment 1"). We set the cost to 1 for each edge insertion and deletion as well as each node insertion and deletion. In our case, edge labels (dis/agreements) are ignored since we evaluate their accuracy via our third metric.

Node-Stance-F1 is a measure of the correctness of the predicted stance of each node towards the root of the tree t. We use this measure instead of directly calculating an F1 score over edge labels. Our reasoning for doing this is that in some cases the predicted graph topology could be different from the gold standard, but nevertheless the stance detection might be correct.

Our models do not directly predict the stance of each node towards the root (topic) node t. Therefore we acquire these stances indirectly by making the simplifying assumption that nodes which support their parent node have the same stance as the parent, and nodes which attack it have the opposite stance (this might be incorrect, for instance, in a case in which one user points out that another's logic is fallacious, while they both share the same opinion towards the main topic). We calculate the macro-F1 score for each graph across two classes: *for* and *against* (micro and macro-averages are not equivalent here because some nodes may be missing in the output; these are put into a third "node missing" class). We then average these macro-F1 scores across all graphs in the test set.

5. Dataset

We sourced our dataset (Table 1) from the *debatabase* website[4]. *Debatabase* is a curated collection of pros and cons on a range of controversial topics across multiple categories such as politics and culture. This data has been used in previous works, notably the args.me corpus [27][5]. We publicly release our own formatted version of this data.[6]

	N. comment threads	Avg. comment thread length (whole word tokens)	Avg. length of all summaries in ASG (whole word tokens)	Avg. comments per thread
Depth-1 trees	590	1365	105	7
Depth-2 trees	590	1054	107	6

Table 1. Statistics of the data collected from *Debatabase*. Depth-1 and Depth-2 trees contain comments on the same topics, but with different tree depths (in Depth-1 trees every node must attack/support the root, in Depth-2 trees, nodes can be linked to the level-1 nodes).

While *Debatabase* is a curated debate collection, and not a repository of comment threads, we chose it because it fits several desirable criteria. Firstly, the comments are tree structured, much like the format of the ASGs which we are interested in generating. Moreover, the support/attack relations between nodes on the tree are implicitly labelled.

[4] `idebate.net/resources/debatabase`. (Note that the online corpus when accessed in June 2024 contained a systematic error in that all "points for" were also included in "points against" across all topics. Our web scraping script automatically removed these misclassified points.)

[5] We cannot use data in that format for our purposes as it has been separated into sentence pairs for training stance classification models; we are instead interested in summarising whole discussions.

[6] `github.com/acidrobin/ASGParsing`

Finally, many of these comments (around 50%) have human-written summaries, which can be used as a gold-standard when training and evaluating an ASG Parsing model.

Using this data, we generate 590 "comment threads", which we define as a multi-party discussion on a single topic. Each of these trees has a corresponding ASG consisting of these same comments summarised with support/attack relations between them labelled. We use an 8:1:1 train-validation-test data split ratio on the *Debatabase* data. The splits are made by topic rather than by data point, such that identical topics do not appear across the three splits.

The data is bound to be far more simplified than a genuine online comments section on a news website or a forum. However, we justify its use here by a lack of publicly-available resources with labelled (dis)agreements, and this simplified dataset is a good starting point for investigation, which can be expanded on in future work.[7]

5.1. Data Format

We store the data in a string format that is amenable to a TANL task framework. An example of this format is shown in Figure 2. The input is a single string consisting of a "main topic" concatenated to a list of comments in the format `"{Comment number}: {comment text}"`. The output is a concatenated list of summaries, with every line corresponding to one of the comments in the input. Each output line has the format `"{Comment number}: ({stance} {Parent number}): {summary text}"`, where "stance" is the stance towards the parent comment, and "parent number" is the number of the parent comment.

Main topic: This House would ban arranged marriages in EU countries
Comment 1: There is simply no feasible way of enforcing laws against arranged marriages, particularly as ...
Comment 2: The new laws can set a precedent, even if it takes time. Bringing into practice such a law...
...
Comment 4: Domestic violence is hardly exclusive to arranged partnerships. Surely focusing exclusively on arranged marriages...

(2a) Sample Input

Comment 1 (attacks **main topic**): It's impossible to police such a law.
Comment 2 (attacks **Comment 1**): The new laws can set a precedent, even if it takes time.
Comment 3 (supports **main topic**): Women in arranged marriages in Europe are disproportionately likely to suffer abuse.
Comment 4 (attacks **Comment 3**): Domestic violence is hardly exclusive to arranged partnerships.

(2b) Sample Output

Figure 2. Example input comment thread and output summary graph from our dataset.

5.2. Depth-1 vs. Depth-2 Trees

We hypothesise that performance will degrade along with the depth of the trees that we attempt to generate. To test this, we created two different task settings: depth-1 trees and depth-2 trees. In the depth-1 tree setting, every comment responds to the main topic, either supporting or attacking it. In the depth-2 setting, comments can also attack these top-level comments (depth-2 support relations are absent in *Debatabase*). We furthermore do not attempt to generate trees deeper than depth-2 due to the absence of these in *Debatabase*.

[7]We do not scrape data from `kialo.com`, arguably an even better source of summary graphs, due to copyright issues; permission to use for research was requested and explicitly denied by the copyright holders.

6. Methodology

In our experiments we compare a baseline described in Section 6.2 against two LLM-based approaches: a two-stage (summarization+stance) pipeline approach (Section 6.3), and a end-to-end TANL approach (Section 6.4).

6.1. *Language Models and fine-tuning process*

The models tested are shown in Table 2. The same hyperparameters and models were used in both the TANL approach and in the two stage pipeline approach (for the latter, we only tested using the same base model for both parts of the pipeline).

One constraint we had when choosing models to evaluate was requiring models that can process a long sequence length (using the Llama-2 tokenizer, the mean average sequence length for this task in the TANL setting was around 2500 tokens, and around 10% of samples had over 4000). Hence, we selected the Longformer model as the most appropriate “small” LLM. We then selected three additional recently-released LLMs in order to benchmark the performance of models with a range of sizes (in number of parameters), namely Phi-2 [28], Llama-2 [29] and GPT-3.5 [30].

Model	Longformer	Phi-2	Llama-2-7b	GPT-3.5
Base model	allenai/led-base-16384	microsoft/phi-2	meta-llama/Llama-2-7b	gpt-3.5-turbo-1106
Params	150M	2.7B	7.9B	175B

Table 2. Models Tested

For GPT-3.5, one epoch of fine-tuning was carried out with the OpenAI API using the default settings; we did not carry out a parameter grid search due to cost constraints. We fine-tuned the other models (Longformer, Phi-2 and Llama-2-7b) on an HPC cluster using two NVIDIA HGX H100 GPUs. For each of these models, we carried out a grid search over the learning rate, within the set $\{1e-3, 1e-4, 1e-5\}$, and of the weight decay parameter within $\{0.001, 0.0\}$. We used 10 epochs of fine-tuning with early stopping.

For Phi-2 and Llama-2-7b, we lacked the computing resources to fine-tune the entire model. We hence used the QLORA (Quantized Low-Rank Adapters) technique [31]. QLORA is a popular parameter-efficient fine-tuning method which involves quantizing the base model and fine-tuning only a small number of Adapter modules which are added to the base model. While training these two models, we also carried out a grid search over the LORA dropout parameter, within the set $\{0.1, 0.5\}$.

6.2. *Baseline*

As a baseline, we used the majority baseline for the classification components combined with a first-sentence baseline for the summarisation component; i.e. we generate a tree where all nodes are children of the root node (the most common parent), and all relations are attack relations (the majority class). Finally, we take the first sentence of each comment as its summary.

6.3. *Pipeline Models*

Fine tuning was done as described in Section 6.1. For the summarisation component, the data used was comments and their summaries; for the stance detection component, comment pairs as the input and relation classes as the output. (see the code [8] for more detail on the specific training classes used and loss functions).

Model	Depth-1 tree			Depth-2 tree		
	ROUGE-2 ↑	Stance-F1 ↑	GED ↓	ROUGE-2 ↑	Stance-F1 ↑	GED ↓
Pipeline Models						
Baseline	0.2263	0.3291	**0.0**	0.3787	0.3247	3.8
Longformer	0.1981	0.1952	0.0	0.1586	0.2461	12.476
Phi-2	0.3042	0.7238	0.0	0.4629	0.2224	12.597
Llama 2	0.3106	0.6931	0.0	0.4541	0.2505	8.25
TANL with fine-tuning						
Longformer	0.1864	0.2217	9.6949	0.1833	0.1943	8.4667
Phi-2	0.3597	0.9093	0.9333	0.546	0.7825	2.5667
Llama 2	**0.3953**	**0.9481**	0.2	**0.5829**	**0.87**	**1.45**
GPT-3.5	0.3137	0.9357	0.5667	0.5422	0.8333	1.85

Table 3. Results of the LLMs tested. Note that up-arrow indicates that higher scores are better, down-arrow indicates that lower scores are better.

Inference in depth-1 tree setting We use a two-class sequence classification model which classifies a pair of texts into either the "attack" or "support" class. To generate the tree at inference time, we iterate over each comment and detect its stance towards the main topic.

Inference in Depth-2 tree setting In the depth-2 tree setting, we use a three-class stance classification model which can classify a child and parent comment into either "support", "attack", or "neutral". We run the classifier over every pair of comments in the input, and use this to build a node-pair matrix for the ASG, which we use as a proxy for an edge probability matrix. We fill in each off-diagonal cell in this matrix with the largest softmax value generated by the LLM for each of the potential link relations, i.e. $MAX(P(support), P(attack))$. We use these max values as a proxy for the model's confidence that a link exists between each pair of nodes. We then run the Chu-Liu Edmonds algorithm [32] over this node-pair matrix to generate a single tree (following the work of other AM researchers such as [33] and more recently [34]). This algorithm is guaranteed to find the spanning tree with the highest weights, which can be interpreted as the set of potential edges with the highest overall model confidence.

6.4. *Post-Processing for TANL models*

We use a simple heuristic algorithm to parse the LLM output and convert it to a tree, using a regular expression. Source code for this algorithm is publicly released.[8] It works by first initialising an output tree with a main topic node t. It then iterates over each line and matches a regex to the syntax of our training data, and builds up the tree incrementally.

7. Results

Our full results are displayed in Table 3. Note that in the flat tree setting, the pipeline models are guaranteed to get a "perfect" GED score of zero because we assume that the flat structure is known *a-priori*. In contrast, the TANL models, which do not have access to this information, sometimes generate an incorrect number of summary nodes.

The TANL approach substantially outperforms the pipeline approach, across all models used and for both depth-1 and depth-2 trees, with the exception of the GED for

[8] `https://github.com/acidrobin/ASGParsing`

flat trees. The explanation for TANL performing better is two-fold: first, error propagation is a known limitation inherent in a pipeline approach. Moreover, the TANL model has access to global information about the debate when making a decision, rather than only having the local information about pairs of comments.

We also observe a degradation in performance between the flat and depth-2 trees in both the Stance-F1 and GED metrics. For the fine-tuned Llama-2 model, degradation in the tree structure corresponds to an average of 1.25 extra nodes misplaced (added or removed) when we add the second level. This seems to indicate a limitation of LLM approaches; that they may struggle when dealing with hierarchical/ compositional structures of increasing depth. These limitations have been observed in other NLP domains, e.g. in syntactic parsing, where tree transformers have been developed to deal with this issue [35]; future work could look at applying similar techniques to ASG parsing.

In general, the performance of the fine-tuned models seem to positively correlate with their size, as we would expect. While the Longformer model failed to beat the baseline, the other three larger models tested scored substantially higher than the baseline across all metrics. One partially surprising result is that the fine-tuned TANL GPT-3.5 slightly under-performed compared to Llama-2, but this may simply be due to the fact that we did not perform a hyper-parameter search on this model, owing to resource constraints.

8. Conclusions

We have proposed a novel task in Argument Mining, ASG Parsing, which involves creating a graphical summary of an argumentative dialogue and have created and made publicly available a dataset that supports this task. We compared two main approaches to this task; a two-stage pipeline approach using separate summarisation and stance detection models, and an end-to-end approach. The end-to-end approach outperformed the pipeline solution across the board, showcasing LLMs' ability to address complex tasks that are intuitively framed as separate tasks. Additionally, we looked at increasing the depth of the summary graphs to a depth-2 tree, and showed that this produces a dip in performance across all models. This result shows that LLMs still struggle with hierarchical structure and that future work should address this limitation. Additionally, the dataset examined in this task was somewhat artificial. Future work should attempt to apply the models examined here to argumentative comment "in the wild".

Acknowledgements

This work was supported by the Centre for Doctoral Training in Speech and Language Technologies (SLT) and their Applications funded by UK Research and Innovation [grant number EP/S023062/1], and by Amazon.

References

[1] Giovanni Paolini, Ben Athiwaratkun, Jason Krone, Jie Ma, Alessandro Achille, Rishita Anubhai, Cicero Nogueira dos Santos, Bing Xiang, and Stefano Soatto. Structured prediction as translation between augmented natural languages. *arXiv:2101.05779*, 2021.

[2] Andrea Calderaro. Social media and politics. In W. Outhwaite and S. Turner, editors, *The SAGE Handbook of Political Sociology*, pages 781–796. SAGE, 2017.

[3] Masayuki Kawarada, Tsutomu Hirao, Wataru Uchida, and Masaaki Nagata. Argument mining as a text-to-text generation task. In *EACL Proceedings (Volume 1: Long Papers)*, pages 2002–2014, 2024.

[4] Kialo. kialo.com, 2024. URL http://www.kialo.com.

[5] James B Freeman. *Argument Structure: Representation and Theory*, volume 18. Springer Science & Business Media, 2011.

[6] Katarzyna Budzynska, Mathilde Janier, Chris Reed, Patrick Saint-Dizier, Manfred Stede, and Olena Yaskorska. A model for processing illocutionary structures and argumentation in debates. In *LREC 2014: Ninth International Conference on Language Resources and Evaluation*, pages 917–924, 2014.

[7] Tatsuki Kuribayashi, Hiroki Ouchi, Naoya Inoue, Paul Reisert, Toshinori Miyoshi, Jun Suzuki, and Kentaro Inui. An empirical study of span representations in argumentation structure parsing. In *ACL Proceedings*, pages 4691–4698, 2019.

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

[9] Isaac Persing and Vincent Ng. End-to-end argumentation mining in student essays. In *NAACL: HLT Proceedings*, pages 1384–1394, 2016.

[10] Steffen Eger, Johannes Daxenberger, and Iryna Gurevych. Neural end-to-end learning for computational argumentation mining. *arXiv:1704.06104*, 2017.

[11] Peter Potash, Alexey Romanov, and Anna Rumshisky. Here's my point: Joint pointer architecture for argument mining. *arXiv:1612.08994*, 2016.

[12] Gaku Morio, Hiroaki Ozaki, Terufumi Morishita, Yuta Koreeda, and Kohsuke Yanai. Towards better non-tree argument mining: Proposition-level biaffine parsing with task-specific parameterization. In *ACL Proceedings*, pages 3259–3266, 2020.

[13] Yuxiao Ye and Simone Teufel. End-to-end argument mining as biaffine dependency parsing. In *Proc. of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 669–678, 2021.

[14] Lang Cao. Autoam: An end-to-end neural model for automatic and universal argument mining. In *International Conference on Advanced Data Mining and Applications*, pages 517–531. Springer, 2023.

[15] Christian Stab and Iryna Gurevych. Identifying argumentative discourse structures in persuasive essays. In *Proc. of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 46–56, 2014.

[16] Andreas Peldszus and Manfred Stede. An annotated corpus of argumentative microtexts. In *Argumentation and Reasoned Action: Proc. of the 1st European Conference on Argumentation, Lisbon*, volume 2, pages 801–815, 2015.

[17] Rob Abbott, Brian Ecker, Pranav Anand, and Marilyn Walker. Internet argument corpus 2.0: An sql schema for dialogic social media and the corpora to go with it. In *Proc. of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 4445–4452, 2016.

[18] John Lawrence and Chris Reed. Aifdb corpora. In *COMMA*, pages 465–466, 2014.

[19] Annette Hautli-Janisz, Zlata Kikteva, Wassiliki Siskou, Kamila Gorska, Ray Becker, and Chris Reed. Qt30: A corpus of argument and conflict in broadcast debate. In *Proceedings of the 13th Language Resources and Evaluation Conference*, pages 3291–3300. European Language Resources Association (ELRA), 2022.

[20] Ramon Ruiz-Dolz, Montserrat Nofre, Mariona Taulé, Stella Heras, and Ana García-Fornes. Vivesdebate: A new annotated multilingual corpus of argumentation in a debate tournament. *Applied Sciences*, 11(15):7160, 2021.

[21] Allen Roush and Arvind Balaji. Debatesum: A large-scale argument mining and summarization dataset. *arXiv:2011.07251*, 2020.

[22] Emma Barker, Monica Lestari Paramita, Ahmet Aker, Emina Kurtić, Mark Hepple, and Robert Gaizauskas. The sensei annotated corpus: Human summaries of reader comment conversations in on-line news. In *Proceedings of the 17th annual meeting of the special interest group on discourse and dialogue*, pages 42–52, 2016.

[23] Ahmet Aker, Emina Kurtic, AR Balamurali, Monica Paramita, Emma Barker, Mark Hepple, and Rob Gaizauskas. A graph-based approach to topic clustering for online comments to news. In *Advances in Information Retrieval: 38th European Conference on IR Research, ECIR 2016, Proceedings 38*, pages 15–29. Springer, 2016.

[24] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81, 2004.

[25] Alberto Sanfeliu and King-Sun Fu. A distance measure between attributed relational graphs for pattern recognition. *IEEE trans. on systems, man, and cybernetics*, (3): 353–362, 1983.

[26] Vladimir I Levenshtein et al. Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet Physics Doklady*, volume 10, pages 707–710, 1966.

[27] Yamen Ajjour, Henning Wachsmuth, Johannes Kiesel, Martin Potthast, Matthias Hagen, and Benno Stein. Data acquisition for argument search: The args.me corpus. In *KI 2019: Advances in Artificial Intelligence: 42nd German Conference on AI, Kassel, Germany, September 23–26, 2019, Proc. 42*, pages 48–59. Springer, 2019.

[28] Mojan Javaheripi, Sébastien Bubeck, Marah Abdin, Jyoti Aneja, Sebastien Bubeck, Caio César Teodoro Mendes, Weizhu Chen, Allie Del Giorno, Ronen Eldan, Sivakanth Gopi, et al. Phi-2: The surprising power of small language models. *Microsoft Research Blog*, 2023.

[29] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv:2307.09288*, 2023.

[30] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[31] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized LLMs. *arXiv:2305.14314*, 2023.

[32] Robert Endre Tarjan. Finding optimum branchings. *Networks*, 7(1):25–35, 1977.

[33] Andreas Peldszus and Manfred Stede. Joint prediction in mst-style discourse parsing for argumentation mining. In *Proc. of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 938–948, 2015.

[34] Jingyuan Huang, Quntian Fang, Sijie Wang, Zhiliang Tian, Feng Liu, Zhen Huang, and Dongsheng Li. Argumentative relationship recognition based on end-to-end multitask learning, 2023. Tentative proceedings of the 12th International Joint Conference on Knowledge Graphs.

[35] Yau-Shian Wang, Hung-Yi Lee, and Yun-Nung Chen. Tree transformer: Integrating tree structures into self-attention. *arXiv preprint arXiv:1909.06639*, 2019.