Computational Modelling of Undercuts in Real-world Arguments

Yuxiao Ye
Tsinghua University
Beijing, China
yeyuxiao@mails.tsinghua.edu.cn

Simone Teufel
University of Cambridge
Cambridge, UK
sht25@cl.cam.ac.uk

Abstract

Argument Mining (AM) is the task of automatically analysing arguments, such that the unstructured information contained in them is converted into structured representations. Undercut is a unique structure in arguments, as it challenges the relationship between a premise and a claim, unlike direct attacks which challenge the claim or the premise itself. Undercut is also an important counterargument device as it often reflects the value of arguers. However, undercuts have not received the attention in the filed of AM they should have — there is neither much corpus data about undercuts, nor an existing AM model that can automatically recognise them. In this paper, we present a real-world dataset of arguments with explicitly annotated undercuts, and the first computational model that is able to recognise them. The dataset consists of 400 arguments, containing 326 undercuts. On this dataset, our approach beats a strong baseline in undercut recognition, with $F_1 = 38.8\%$, which is comparable to the performance on recognising direct attacks. We also conduct experiments on a benchmark dataset containing no undercuts, and prove that our approach is as good as the state of the art in terms of recognising the overall structure of arguments. Our work pioneers the systematic analysis and computational modelling of undercuts in real-world arguments, setting a foundation for future research in the role of undercuts in the dynamics of argumentation.

1 Introduction

Social media allows people to express divergent opinions on the same subject and to reach many more people than was possible in earlier times. However, the ubiquity of the internet and social media also has some negative consequences. One of these is the growing polarisation between individuals holding different beliefs and opinions. It is thus increasingly important to promote productive communication and understanding among people with opposing perspectives. This is where Argument Mining (AM) comes into play. AM aims to automatically identify and extract arguments from natural language texts (Peldszus and Stede, 2013; Green et al., 2014). It can convert unstructured textual information into structured argument data, which not only identifies the argumentative text segments in the text but also the relations between them (Prakken and Vreeswijk, 2002; Lawrence and Reed, 2020).

A critical aspect of AM is recognising and understanding various argumentative structures, including undercuts. An undercut challenges the relationship between a premise and a claim (Pollock, 1987), unlike direct attacks that challenge the claim or the premise itself. Due to its complex structure, it is difficult to annotate undercuts or to computationally model them. There exist some AM datasets with annotation of undercuts (Peldszus and Stede, 2015a; Visser et al., 2020), but they are often limited in the size, the quality of source text, or the annotation scheme. To our best knowledge, there is no existing AM models that can automatically recognise undercuts.

To addresses this gap, in this paper we present a novel dataset of real-world arguments from Quora[1], a popular question-answering platform. Our dataset consists of 400 arguments, including 326 explicitly annotated undercuts, making it the largest AM dataset with such annotations to date. We also develop the first computational approach capable of recognising undercuts, proposing an innovative undercut-inclusive dependency representation and a GNN-based neural dependency parser.

Our work contributes to the field of AM in several ways. Firstly, we provide a comprehensive dataset with detailed annotations of undercuts, offering a valuable resource for future research. Secondly, our undercut-inclusive representation allows

[1] https://www.quora.com

Proceedings of the 11th Workshop on Argument Mining, pages 59–68

existing neural dependency parsers to process undercuts effectively, preserving their unique status within argument structures. Lastly, our experimental results demonstrate that our GNN-based parser outperforms existing biaffine parsers in recognising undercuts and maintains state-of-the-art performance in the general AM task.

2 Related Work

Our work is closely related to existing approaches to AM, and the studies on undercuts in the field of AM.

2.1 AM Approaches

There are two kinds of approaches to AM in general, pipelined and end-to-end. Pipelined approaches break down AM into several subtasks and process them sequentially, such as in Persing and Ng (2016), Mayer et al. (2020), and Ruiz-Dolz et al. (2021). End-to-end approaches to argument mining allow for the prediction of the full argument structure with a single model, and have been gaining popularity due to their advantages over pipelined approaches, including avoiding error propagation and eliminating the need for designing different models for different subtasks (Ye and Teufel, 2021).

The tree or graph structure of arguments enables some end-to-end approaches to formulate argument mining as a dependency parsing problem. For example, Morio et al. (2020) use bidirectional LSTMs (BiLSTMs) (He et al., 2016) to encode argument components, and a biaffine dependency parser (Dozat and Manning, 2018) to classify components and their relations. Bao et al. (2021) propose a neural transition-based model to predict the dependency structure of arguments. These approaches all assume that the input text is already segmented.

In contrast, Eger et al. (2017) and Ye and Teufel (2021) take raw text as input in their end-to-end approaches based on dependency parsing. Ye and Teufel (2021) report better results than those by Eger et al. (2017), crediting the improvement to their token-level dependency representation of argument and the biaffine dependency parser they use. Our approach is based on the work by Ye and Teufel (2021), except that ours can computationally model undercuts.

2.2 Studies on Undercuts

Undercuts play a critical role in challenging the soundness or validity of an argument. They have been well defined in various theoretical argumentation models, such as Pollock's argumentation model (Pollock, 1987) and Besnard and Hunter's argumentation model (Besnard and Hunter, 2009).

In the field of AM, some datasets based on the Argument Interchange Format (Chesnevar et al., 2006) may include undercuts, for example, the QT30 corpus by Hautli-Janisz et al. (2022). However, their inclusion of undercuts is incidental and will form a small part of the overall dataset.

The Microtext dataset (Peldszus and Stede, 2015a) is one of the few AM datasets that contain explicitly annotated undercuts. The source text was produced in a highly controlled text generation experiment. The size of this dataset is small (7,846 tokens in total), each document only containing about five segments. Also, its creators do not categorise these undercuts or provide any automatic method to recognise them in their follow-up experiments. Mim et al. (2022) present some ideas that touch upon the phenomenon of undercuts in their dataset, although they do not explicitly mention undercuts when doing so. But nobody has studied which kinds of undercut strategies exist, nor is there large corpus data about them available. In contrast, to the best of our knowledge, the QuoraAM dataset we present is the biggest AM dataset that contains annotation of undercuts. We provide a taxonomy of undercuts after manually examining the annotation of our dataset, along with its distribution in our dataset. Moreover, we propose the first approach to computationally modelling undercuts.

3 The QuoraAM Dataset

We collected 400 arguments from Quora, and named this dataset as the QuoraAM dataset.

Compared to user-generated content on other online discussion platforms such as ChangeMyView, Kialo, idebate.org, and Twitter, arguments on Quora are more in line with our research interest. On Quora, users can present detailed and well-reasoned points of view in their answers to a question. As a result, each answer on Quora can be seen as containing a stand-alone cogently structured complete argument, often supplemented with explanations and supporting evidence. Platforms including ChangeMyView, Kialo and idebate.org are specifically designed for interactive debates and

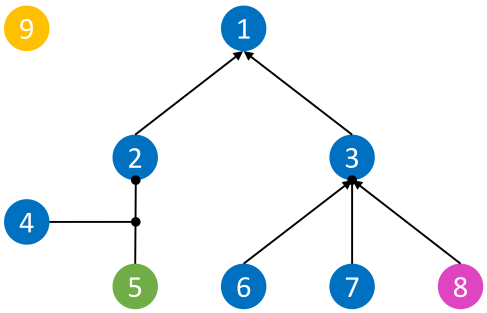

Figure 1: An example argument graph using our annotation scheme. Nodes in different colours represent components of different categories. Edges represent relations: arrow-head = SUPPORT, circle-head = ATTACK.

discussions, with a focus on the process of changing one's mind through constructive conversations. Each post on such platforms may only contain an incomplete argument, and intertextual referencing is frequent among posts, making the analysis of such posts more difficult. Twitter, unlike those previous platforms, is a more open and informal platform, where arguments can take on a more combative tone. As a result, arguments there may not always be rational and often lack the depth and nuance seen on the other platforms. Additionally, due to Twitter's character limit, arguments in tweets tend to be very short, with structures that may be too simplistic to warrant a detailed analysis. Therefore, we chose Quora over other online discussion platforms.

We first manually pre-segmented the QuoraAM dataset, and then trained two annotators to annotate this dataset using our annotation scheme. This dataset will be used for our experiments in this paper. To the best of our knowledge, this is the biggest argument mining dataset that contains explicit annotation of undercuts.

3.1 Annotation Scheme

Our scheme is illustrated in Figure 1. It includes four argument component categories and two argument relation types.

The unit of annotation for components is a segment that can be part of a sentence, a sentence or a sequence of sentences. We define four component categories in our annotation scheme, including **PROPOSITION**, **STAKE**, **ANECDOTE**, and **ANALOGY**. These categories are decided based on our manual observation of arguments on Quora and the argumentation schemes by Walton et al. (2008).

Our scheme only has two relation categories, namely **SUPPORT** (*e.g.* component 2 supporting component 1) and **ATTACK** (*e.g.* component 7 attacking component 3). In our scheme, the representation of undercuts does not rely on relation labels, but on the target of an attacking relation: if the target is a relation, an undercut occurs (*e.g.* component 4 undercutting the relation between component 5 and component 2); otherwise it is just a typical direct SUPPORT.

Component 9 in Figure 1 is a STAKE, and is a stand-alone component, In our scheme, STAKEs are always stand-alone components, which do not hold any relation to other components or relations.

3.2 Dataset Creation

In order to collect 400 answers from Quora, we first selected the first 20 topics in the topic catalogue on Kialo[2] (*e.g.* "Politics", "Society", and "Technology"). Under each topic, we identified the top five popular questions, resulting in 100 candidate questions (5 questions per topic). These Kialo questions were then used to search for corresponding questions on Quora.

In the second step, we aimed to select one relevant question on Quora for each topic. We used the full string of each Kialo question as a query on Quora. Suitable questions were those relevant to the Kialo query and with at least 50 answers. If no suitable question was found, we refined our search using key terms from the Kialo question. If we still could not find a suitable Quora question, we proceeded to the next Kialo question. This process yielded 20 selected Quora questions.

In the third step, we chose 20 answers for each selected Quora question. A qualified answers must: 1) directly address the topic; 2) contain at least 30% argumentative material; and 3) be between 60 and 800 wordpieces after being tokenised by the BERT WordPiece tokeniser (Wolf et al., 2020).

Using this data collection method, we selected 400 (20×20) answers that cover various topics such as politics, environment, education, and equality. Since arguments on Quora happen in a question-answering context, each answer was appended to its corresponding question, forming a "document" for our dataset.

We manually segmented the raw text in the QuoraAM dataset and trained two annotators to apply our annotation scheme. The two annotators first

[2] https://www.kialo.com. We turned to Kialo for topic selection because Quora does not provide such a catalogue.

annotated the same 60 documents for the annotation study. The remaining 340 documents were randomly split into two equal subsets. Each subset was assigned to one annotator for annotation. The annotation process took approximately 80 hours per annotator. To compile the final collection of 400 annotated documents, we randomly selected one of the doubly annotated documents from the first step and combined it with the 340 documents from the second step.

Inter-annotator agreement was measured using Krippendorff's alpha (Krippendorff, 2018) for component classification and graph similarity measures (Kirschner et al., 2015; Putra et al., 2022) for relation identification. For component classification, the score of Krippendorff's alpha is $\alpha = 0.78$ ($N = 7,883, n = 2, k = 5$). According to the interpretation scale in Krippendorff (2018), this score is acceptable for "drawing tentative conclusions" ($\alpha \geq 0.67$), and is close to the threshold ($\alpha \geq 0.80$) for being considered "reliable". For relation identification, the graph similarity scores are: $\text{Kirschner}^{\text{mean}} = 0.69$, $\text{Kirschner}^{\text{f}} = 0.67$, $\text{MAR}^{\text{link}} = 0.64$, $\text{MAR}^{\text{path}} = 0.54$, and $\text{MAR}^{\text{dSet}} = 0.84$.

3.3 Dataset Statistics

Table 1 shows the statistics of the QuoraAM dataset. The dataset contains a total of 118,573 tokens, which is much larger than the Microtext dataset (7,846 tokens). There are over 7,800 segments in the dataset, with approximately 56% being argumentative. The QuoraAM dataset as distributed is randomly divided into three subsets: 280 documents for training, 40 for development, and 80 for testing.

Table 2 shows the distribution of components and relations. The dataset includes over 4,000 PROPOSITIONS, around 200 ANALOGIES, 79 ANECDOTES, and 28 STAKES. Due to the small number of STAKES, we merged them with ANECDOTES in our experiments in Section 5, though the original categories are preserved in the dataset for future research.

	All	Per document
Token	118,573	296.4
Segment	7,883	19.7
Component	4,381	11.0
Sentence	6,398	16.0
Paragraph	2,826	7.1

Table 1: Statistics of the QuoraAM dataset.

		All	Per doc
Component	PROPOSITION	4,075 (51.7%)	10.2
	STAKE	28 (0.4%)	0.1
	ANECDOTE	79 (1.0%)	0.2
	ANALOGY	199 (2.5%)	0.5
	Total	4,381	11.0
	Non-arg	3,502 (44.4%)	8.8
Relation	SUPPORT	2,752 (69.8%)	6.9
	ATTACK	1,190 (30.2%)	3.0
	Normal attack	864 (72.6%)	2.2
	Undercut	326 (27.4%)	0.8
	Total	3,942	9.9

Table 2: Component and relation distribution of the QuoraAM dataset.

In terms of relations, the QuoraAM dataset includes 2,752 instances of SUPPORT and 1,190 instances of ATTACK. There are 326 undercuts in the dataset, which constitutes approximately 27% of all ATTACKs. This result confirms the prevalence of undercuts in Quora arguments.

3.4 Categories of Undercuts

We manually examined all undercuts in the QuoraAM dataset, classifying them into three categories:

- **Rejection**: Rejecting the relation by denying the relevance between the source component and the target component.
- **Low importance**: Questioning the importance of the relation, or providing more important reasons.
- **Alternative option**: Stating that the current solution is not the only option, or providing alternative options. This kind of undercuts often appears in arguments about policies.

Figure 2 shows the distribution of undercuts in the QuoraAM dataset. "Low importance" (41%) is the most frequent, followed by "Alternative option" (36%), "Rejection" (15%), and others (8%). This indicates that Quora authors prefer less direct methods of undercutting relations, often pointing out weaknesses or suggesting alternatives rather than outright rejection.

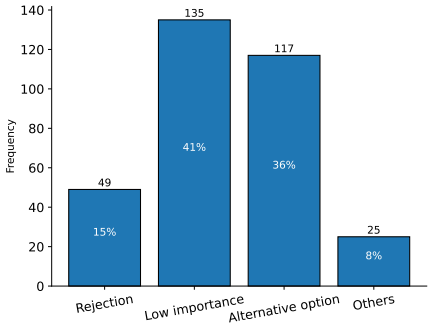

Figure 2: Distribution of undercuts in the QuoraAM dataset.

4 Computational Modelling of Undercuts

Through a redesign of an existing dependency representation of arguments by Ye and Teufel (2021), we are able to directly use existing neural dependency parsers to computationally model undercuts. We also apply a modified GNN-based dependency parser for improved performance.

4.1 Undercut-inclusive Dependency Representation

Following Ye and Teufel (2021), we also frame AM as a dependency parsing task. They propose a token-level dependency representation for arguments in order to approach argument mining in an end-to-end fashion. However, their dependency representation cannot be used for arguments containing undercuts, as undercuts involve relations between a component and a relation, and a relation cannot be a dependent or a head in typical dependency representations. Therefore, we designed an undercut-inclusive dependency representation for arguments (shown in Figure 3) to allow existing neural parsers to process undercuts directly. It is a modification of the undercut-exclusive representation by Ye and Teufel (2021). This figure uses the category labels in the Persuasive Essays dataset (Stab and Gurevych, 2017) to be consistent with the figure in Ye and Teufel (2021). New features of our undercut-inclusive representation are as follows:

- A relation node (shown as dashed-line nodes in Figure 3) for each token in the argument is added. Each relation node is indexed with the token number of its corresponding token, followed by a prime. The relation nodes are meant to represent relations. This is very different from the undercut-exclusive representation, where relations are represented by edges.

- The relation node (*e.g.* relation node $7'$) of the last token (token 7) in a component is always the head of that token, and represents the relation from that component to its target, or the other way around. The edge label between the last token in a component and its corresponding relation node is written as ($segment\_label$, REL), where REL means "relation". For example, the label of the edge between token 7 and relation node $7'$ is (P, REL).

- If the relation is a SUPPORT or a direct ATTACK, the relation node's outgoing edge points to the last token in the source component. Its incoming edge comes from the last token of the target component. For example, the fact that "it killed much marine life" (tokens 3-7) supports "tourism has threatened nature" (tokens 9-12) is expressed by the incoming edge of relation node $7'$ from token 12, and the outgoing edge of relation node $7'$ to node 7.

- If the relation is an undercut, the relation node's outgoing edge points to the last token in the undercutting component. The incoming edge comes from another relation node, rather than the last token in a component. For example, the fact that tokens 14-15 undercuts the relation between tokens 3-7 and tokens 9-12 is expressed by the incoming edge of relation node $15'$ from relation node $7'$, and the outgoing edge of relation node $15'$ to node 15.

This design treats all relations as nodes, enabling undercuts to be modelled as relations between nodes while preserving their unique status. In this way, existing neural dependency parsers are able to process undercuts directly.

4.2 GNN-based Neural Dependency Parser

We modified the GNN-based dependency parser proposed by Ji et al. (2019), which uses Graph Attention Networks (GANs) (Veličković et al., 2017) to model higher-order dependencies. Compared to the biaffine parser used by Ye and Teufel (2021), we expect our GNN-based parser to capture global argument structure and higher-order dependencies more effectively.

The mathematical description of our GNN-based parser is as follows:

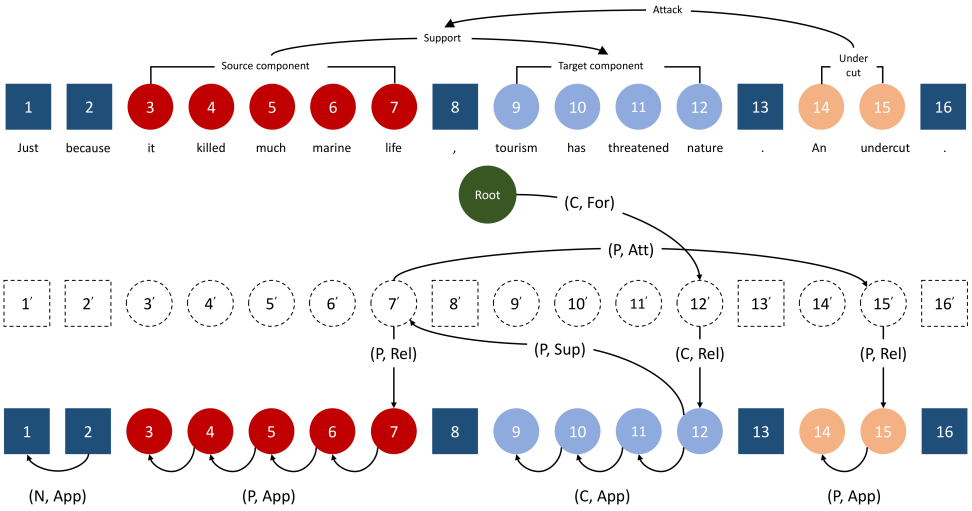

Figure 3: The structure of an example argument with a pseudo undercutting component (written as "An undercut"), and its undercut-inclusive dependency representation. C = CLAIM, P = PREMISE, Sup = SUPPORT, N = non-argumentative, App = append, Rel = relation.

$$r_S = \text{BERT}(s_1 s_2 ... s_n) \quad (1)$$

$$r_{ROOT} = \text{FFN}^{\text{ROOT}}(\text{mean}(r_S), \text{axis}=0) \quad (2)$$

$$R = [r_S; r_{ROOT}], \text{axis}=0 \quad (3)$$

$$H^{\text{e\_h}}, H^{\text{l\_h}}, H^{\text{e\_d}}, H^{\text{l\_d}} = \text{FFN}(R) \quad (4)$$

$$H_{\text{r}}^{\text{e\_h}}, H_{\text{r}}^{\text{l\_h}}, H_{\text{r}}^{\text{e\_d}}, H_{\text{r}}^{\text{l\_d}} = \begin{cases} \text{FFN}^{\text{rel\_node}}(R) & \text{if undercut\_inclusive} \\ \emptyset & \text{if undercut\_exclusive} \end{cases} \quad (5)$$

$$H_{\text{G}}^{\text{e\_h}}, H_{\text{G}}^{\text{l\_h}}, H_{\text{G}}^{\text{e\_d}}, H_{\text{G}}^{\text{l\_d}} = \text{GNN}_{\text{layer}=2}(([H^{\text{e\_h}}; H^{\text{l\_h}}], \text{axis}=1), ([H^{\text{e\_d}}; H^{\text{l\_d}}], \text{axis}=1), ([H_{\text{r}}^{\text{e\_h}}; H_{\text{r}}^{\text{l\_h}}], \text{axis}=1), ([H_{\text{r}}^{\text{e\_d}}; H_{\text{r}}^{\text{l\_d}}], \text{axis}=1)) \quad (6)$$

$$\text{Biaff}(x, y) = x^{\top} \text{U} y + \text{W}(x \oplus y) + b \quad (7)$$

$$sc^{\text{edge}} = \text{Biaff}^{\text{edge}}(H_{\text{G}}^{\text{e\_h}}, H_{\text{G}}^{\text{e\_d}}) \quad (8)$$

$$sc^{\text{label}} = \text{Biaff}^{\text{label}}(H_{\text{G}}^{\text{l\_h}}, H_{\text{G}}^{\text{l\_d}}) \quad (9)$$

$$y_{i,j}^{\prime\text{edge}} = \{sc_{i,j}^{\text{edge}} \geq 0\} \quad (10)$$

$$y_{i,j}^{\prime\text{label}} = \arg\max \ sc_{i,j}^{\text{label}} \quad (11)$$

We calculate four representations for all tokens $(H^{\text{e\_h}}, H^{\text{l\_h}}, H^{\text{e\_d}}, H^{\text{l\_d}})$ in Equations 1-4. Relation nodes' representations $(H_{\text{r}}^{\text{e\_h}}, H_{\text{r}}^{\text{l\_h}}, H_{\text{r}}^{\text{e\_d}}, H_{\text{r}}^{\text{l\_d}})$ are produced in Equation 5 when necessary.

In Equation 6, the eight representations above are fed into a two-layer GNN encoder, forming a fully connected graph. The head representation and the dependant representation of each node are concatenated to form a general representation. The general representation of each node is then aggregated and updated through the GNN layers. A detailed explanation can be found in Ji et al. (2019).

The GNN-encoded representations are then used for edge and label prediction, as described in Equations 7-11.

The final loss is calculated in the same way as that in Ji et al. (2019).

5 Experiments

We conducted two experiments in order to test:

- **Experiment 1** – the effectiveness of our undercut-inclusive representation and GNN-based parser on undercut recognition;
- **Experiment 2** – the impact of our undercut-inclusive representation on arguments without undercuts.

Results from these experiments can also be used to test the effectiveness of our GNN-based parser on AM in general.

5.1 Datasets

In Experiment 1, we used our QuoraAM dataset, processing each document as a data point to capture argument relations spanning paragraphs. This is because relations in the QuoraAM dataset are more likely to span across paragraphs, rather than

operating within individual paragraphs, as paragraph breaks are often not used consistently in this dataset. We merged all instances of STAKE and ANECDOTE in the QuoraAM dataset to form a new category called STAKE+ANECDOTE, as discussed in Section 3.3.

In Experiment 2, we used the Persuasive Essays dataset, which is a benchmark constructed by Stab and Gurevych (2017). This dataset is also used in Ye and Teufel (2021). It comprises 402 persuasive essays randomly selected from an online forum, with 322 essays used for training and 80 essays for testing. Considering that most relations hold within paragraphs in this dataset, each paragraph was processed as a separate argument, aligning with Ye and Teufel (2021).

In both experiments, we performed the same post-processing steps as Ye and Teufel (2021).

5.2 Systems

In Experiment 1, since no existing approaches model undercuts computationally, we built a new baseline model *Biaff_exc_r*, using the biaffine parser in Ye and Teufel (2021) with their undercut-exclusive representation. We simulate Peldszus and Stede (2015b)'s approach by transforming all undercuts into direct ATTACKs during training. During inference, we randomly convert all predicted direct ATTACKs to undercuts in proportion to the ratio of undercuts to overall ATTACKs in the original QuoraAM dataset.

In Experiment 2, we selected two baseline models: *Biaff_exc* (the *BiPAM* model in Ye and Teufel (2021)), the biaffine parser in Ye and Teufel (2021) with their undercut-exclusive representation; and *GNN_exc*, our GNN-based parser with the undercut-exclusive representation in Ye and Teufel (2021).

We implemented two models in the two experiments to compare with the baselines: *Biaff_inc*, the biaffine parser with our undercut-inclusive representation; and *GNN_inc*, our GNN-based parser with our undercut-inclusive representation.

6 Results and Discussion

Table 3 shows the F_1 scores for component and relation identification in Experiment 1. Table 4 shows the results for Experiment 2. We used permutation tests from Ye and Teufel (2021) to test the significance of our results.

In Experiment 1, for undercut recognition, both models significantly outperform the baseline (*i.e.* *Biaff_exc_r*) by a large margin. *Biaff_inc* outperforms the baseline by 15.8% ($Biaff\_inc = 21.6\%$, $baseline = 5.8\%$, $p < 0.01$), which suggests that our undercut-inclusive representation can significantly improve undercut recognition compared to the undercut-inclusive representation by Ye and Teufel (2021). The increase is 33.0% for *GNN_inc* ($GNN\_inc = 38.8\%$, $baseline = 5.8\%$, $p < 0.01$). The performance on undercut recognition of *GNN_inc* is comparable to that on recognising direct ATTACKs ($Undercut = 38.8\%$, $DirectAttack = 40.3\%$). This suggests that undercuts, despite their intricate nature, are structures that can be effectively recognised using our approach. It also shows that the GNN-based parser is better than the biaffine parser at recognising undercuts: *GNN_inc* significantly outperforms *Biaff_inc* by 17.2% ($GNN\_inc = 38.8\%$, $Biaff\_inc = 21.6\%$, $p < 0.01$).

The results from two experiments suggest that the GNN-based parser is more efficient than the biaffine parser. In Experiment 1, *GNN_inc* significantly outperforms Biaff_inc by 3.8% for component identification ($GNN\_inc = 66.2\%$, $Biaff\_inc = 62.4\%$, $p < 0.01$) and 4.1% for relation identification ($GNN\_inc = 45.8\%$, $Biaff\_inc = 41.7\%$, $p < 0.01$); in Experiment 2, *GNN_inc* significantly outperforms Biaff_inc by 1.0% for component identification ($GNN\_inc =$

	Biaff_exc_r	Biaff_inc	GNN_inc
Component	62.4	62.4	66.2
Relation	35.2	41.7	45.8
SUPPORT	45.2	45.1	48.0
ATTACK	11.7	33.8	39.9
Direct ATTACK	14.0	38.4	40.3
Undercut	5.8	21.6	38.8

Table 3: F_1 scores for models on the QuoraAM dataset in Experiment 1.

Model	Component	Relation
Biaff_exc	72.9	45.9
Biaff_inc	72.8	45.9
GNN_exc	73.8	49.4
GNN_inc	73.8	49.4

Table 4: F_1 scores for models on the Persuasive Essays dataset in Experiment 2.

73.8%, $Biaff\_inc = 72.8\%$, $p < 0.01$) and 3.5% for relation identification ($GNN\_inc = 49.4\%$, $Biaff\_inc = 45.9\%$, $p < 0.01$).

Regarding the comparison between the undercut-exclusive and the undercut-inclusive representations in Experiment 2, the results reveals no significant difference in performance. Both representations yield similar F_1 scores for component and relation identification, which implies that our undercut-inclusive representation can also be used for arguments containing no undercuts, without performance compromise. This result suggests the flexibility of our undercut-inclusive representation.

6.1 Recognising undercuts: biaffine parser *vs.* GNN-based parser

To understand the disparity between the biaffine parser and the GNN-based parser in recognising undercuts, we compared the errors made by both parsers, so that we can discern which parts of an undercut are most error-prone for the GNN-based parser.

Figure 4 shows the structure of an undercut with its dependency representation. Nodes 1-3 are components, with nodes 1' and 2' as their corresponding relation nodes. An undercut has three elements: an undercutting component (*e.g.* node 1 in Figure 4), its target relation ("node 2 ⇐ node 2' ⇐ node 3"), and the link between them ("node 1' ⇐ node 2'"). Errors can occur in recognising any single element or a combination of them. We focused on three error types illustrated in Figure 4: type I (errors in recognising the undercutting component); type II (errors in recognising the target relation); and type III (errors only in recognising the link between them). Please note that type III only include the cases where the link is incorrectly predicted but the undercutting component and the target relation are correctly predicted.

We performed an error analysis by counting the errors made by our models. Figure 5 shows the error distribution among all undercuts (N=58) in the test set. Due to the small sample size, we refrain from testing the statistical significance of the results in this figure, and instead interpret the numbers only qualitatively. According to Figure 5, *Biaff_inc* produces relatively 20% more type I errors and 26% more type II errors than *GNN_inc*. For type III errors, the difference becomes much bigger, which is 300%. This pattern suggests that the GNN-based parser recognises all three elements of an undercut more effectively, especially the link between the undercutting component and the target relation.

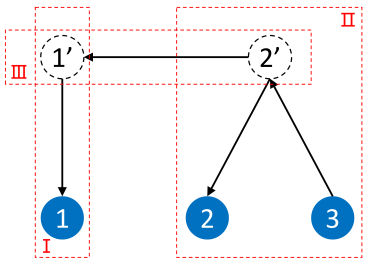

Figure 4: The dependency representation of an undercut, and three types of error in undercut recognition.

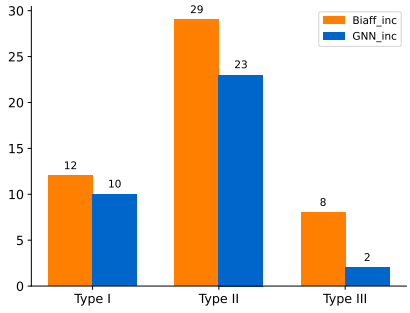

Figure 5: Distribution of three types of error in undercut recognition.

Regarding the disparity in link recognition between the GNN-based parser and the biaffine parser, we hypothesise that the GNN-parser's advantage in identifying higher-order dependencies is crucial. Figure 6 illustrates the number of hops required for different relation representations: (a) in the undercut-exclusive representation, a direct ATTACK requires one hop; (b) in the undercut-inclusive representation, a direct ATTACK requires two hops; (c) in the undercut-inclusive representation, an undercut requires three hops.

A direct ATTACK is a 1-hop relation in the undercut-exclusive representation, but becomes a 2-hop relation in the undercut-inclusive representation. Despite this, performance on the Persuasive Essays dataset shows no significant loss, imply-

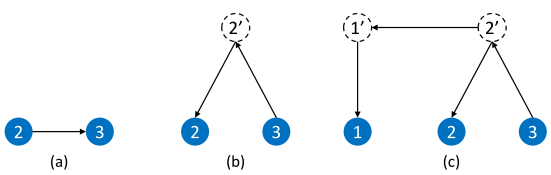

Figure 6: Illustration of the number of hops required for different relation representations.

ing neither parser suffers from the transition from 1-hop to 2-hop relations.

However, when comparing direct ATTACKs and undercuts in the undercut-inclusive representation (2-hop *vs*. 3-hop relations), both models show weaker performance for undercuts. According to Table 3, the F_1 score of *Biaff_inc* for undercuts lags by 16.8% compared to direct ATTACKs, while for *GNN_inc* the difference is only 1.5%. This suggests the GNN-based parser handles the increase from two to three hops better than the biaffine parser, supporting our earlier prediction that the GNN-based parser excels at capturing higher-order dependencies.

7 Conclusion

In this study, we addressed a critical gap in AM by focusing on the computational recognition of undercuts, a complex yet essential structure in arguments. Existing AM research has largely overlooked undercuts, primarily due to the lack of annotated datasets and effective computational models.

To tackle this, we introduced a novel dataset sourced from Quora. This dataset, is the largest that contains undercuts, providing a valuable resource for future AM research. We also developed the first computational approach capable of recognising undercuts, featuring an undercut-inclusive dependency representation and a GNN-based neural dependency parser.

Our experiments indicated that our undercut-inclusive representation can be effectively used for undercut recognition, and it does not compromise performance on datasets without undercuts, showcasing its flexibility and robustness. The results also demonstrated that the GNN-based parser is effective in general AM as well as in recognising undercuts. The GNN-based parser's ability to capture higher-order dependencies was evident, showing a notable advantage in accurately identifying the intricate structures of undercuts.

References

Jianzhu Bao, Chuang Fan, Jipeng Wu, Yixue Dang, Jiachen Du, and Ruifeng Xu. 2021. A neural transition-based model for argumentation mining. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6354–6364.

Philippe Besnard and Anthony Hunter. 2009. Argumentation based on classical logic. *Argumentation in artificial intelligence*, pages 133–152.

Carlos Chesnevar, Sanjay Modgil, Iyad Rahwan, Chris Reed, Guillermo Simari, Matthew South, Gerard Vreeswijk, Steven Willmott, et al. 2006. Towards an argument interchange format. *The knowledge engineering review*, 21(4):293–316.

Timothy Dozat and Christopher D Manning. 2018. Simpler but more accurate semantic dependency parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 484–490.

Steffen Eger, Johannes Daxenberger, and Iryna Gurevych. 2017. Neural end-to-end learning for computational argumentation mining. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11–22.

Nancy Green, Kevin D Ashley, Diane Litman, Chris Reed, and Vern Walker. 2014. Proceedings of the first workshop on argumentation mining. In *Proceedings of the First Workshop on Argumentation Mining*.

Annette Hautli-Janisz, Zlata Kikteva, Wassiliki Siskou, Kamila Gorska, Ray Becker, and Chris Reed. 2022. 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).

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Tao Ji, Yuanbin Wu, and Man Lan. 2019. Graph-based dependency parsing with graph neural networks. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2475–2485.

Christian Kirschner, Judith Eckle-Kohler, and Iryna Gurevych. 2015. Linking the thoughts: Analysis of argumentation structures in scientific publications. In *Proceedings of the 2nd Workshop on Argumentation Mining*, pages 1–11.

Klaus Krippendorff. 2018. *Content analysis: An introduction to its methodology*. Sage publications.

John Lawrence and Chris Reed. 2020. Argument mining: A survey. *Computational Linguistics*, 45(4):765–818.

Tobias Mayer, Elena Cabrio, and Serena Villata. 2020. Transformer-based argument mining for healthcare applications. In *ECAI 2020*, pages 2108–2115. IOS Press.

Farjana Sultana Mim, Naoya Inoue, Shoichi Naito, Keshav Singh, and Kentaro Inui. 2022. Lpattack: A feasible annotation scheme for capturing logic pattern of attacks in arguments. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 2446–2459.

Gaku Morio, Hiroaki Ozaki, Terufumi Morishita, Yuta Koreeda, and Kohsuke Yanai. 2020. Towards better non-tree argument mining: Proposition-level biaffine parsing with task-specific parameterization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3259–3266.

Andreas Peldszus and Manfred Stede. 2013. From argument diagrams to argumentation mining in texts: A survey. *International Journal of Cognitive Informatics and Natural Intelligence (IJCINI)*, 7(1):1–31.

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

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

Isaac Persing and Vincent Ng. 2016. End-to-end argumentation mining in student essays. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1384–1394.

John L Pollock. 1987. Defeasible reasoning. *Cognitive science*, 11(4):481–518.

Henry Prakken and Gerard Vreeswijk. 2002. Logics for defeasible argumentation. *Handbook of philosophical logic*, pages 219–318.

Jan Wira Gotama Putra, Simone Teufel, and Takenobu Tokunaga. 2022. Annotating argumentative structure in english-as-a-foreign-language learner essays. *Natural Language Engineering*, 28(6):797–823.

Ramon Ruiz-Dolz, Jose Alemany, Stella M Heras Barberá, and Ana García-Fornes. 2021. Transformer-based models for automatic identification of argument relations: a cross-domain evaluation. *IEEE Intelligent Systems*, 36(6):62–70.

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

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903*.

Jacky Visser, Barbara Konat, Rory Duthie, Marcin Koszowy, Katarzyna Budzynska, and Chris Reed. 2020. Argumentation in the 2016 us presidential elections: annotated corpora of television debates and social media reaction. *Language Resources and Evaluation*, 54(1):123–154.

Douglas Walton, Christopher Reed, and Fabrizio Macagno. 2008. *Argumentation schemes*. Cambridge University Press.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

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