EDU-AP: Elementary Discourse Unit based Argument Parser

Sougata Saha, Souvik Das, Rohini Srihari
State University of New York at Buffalo
Department of Computer Science and Engineering
`{sougatas, souvikda, rohini}@buffalo.edu`

Abstract

Neural approaches to end-to-end argument mining (AM) are often formulated as dependency parsing (DP), which relies on token-level sequence labeling and intricate post-processing for extracting argumentative structures from text. Although such methods yield reasonable results, operating solely with tokens increases the possibility of discontinuous and overly segmented structures due to minor inconsistencies in token level predictions. In this paper, we propose EDU-AP, an end-to-end argument parser, that alleviates such problems in dependency-based methods by exploiting the intrinsic relationship between elementary discourse units (EDUs) and argumentative discourse units (ADUs) and operates at both token and EDU level granularity. Further, appropriately using contextual information, along with optimizing a novel objective function during training, EDU-AP achieves significant improvements across all four tasks of AM compared to existing dependency-based methods.

1 Introduction

Considered an integral mode of persuasion, argumentation is prevalent in our daily verbal communication and represents chains of thought patterns and reasoning. An argument constitutes claims and premises, with the claim being the central controversial statement of the argument, and the premise either supporting or attacking the claim by providing the reasoning for the claim (Stab and Gurevych, 2014b). Argument mining (AM) is a recent research field in computational linguistics, that deals with analyzing discourse on the pragmatics level, and finding argumentation structures in natural language texts (Mochales and Moens, 2011; Lippi and Torroni, 2016; Lawrence and Reed, 2019). AM comprises four sub-tasks: (a) *text segmentation:* identifying ADUs from text, by separating argumentative units from non-argumentative units; (b) *component classification:* associating each identified ADU with a tag from a pre-defined labeling scheme (e.g., Claim or Premise); (c) *relation detection:* determining if any relationship exists between pairs of ADUs; (d) *relation classification:* labeling a determined relationship with a tag from a pre-defined labeling scheme (e.g., attack or support). When performed successfully, AM generally leads to the creation of an *argumentation graph* (AG) (Peldszus and Stede, 2013): a graphical framework for representing arguments, whereby nodes represent claims and premises, and the edges represent diverse relationships (e.g., support, attack) between arguments. Such graphical structures not only help analyze discourse but also aids in the creation of dialogue agents that can leverage the AGs for response generation (Chalaguine and Hunter, 2020; Slonim et al., 2021). Figure 1 illustrates such a graphical relationship, where the premise "biking reduces stress and releases endorphins", supports the claim "biking is good".

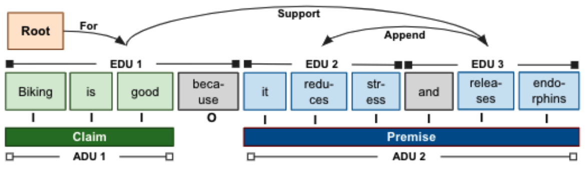


Figure 1: Dependency tree for the argument "Biking is good because it reduces stress, and releases endorphins".

With an increased interest in engendering purposeful and persuasive conversational agents, the need for argument parsers that can automatically and effectively extract, parse and relate argumentative components end-to-end from natural language text is on the rise. In this paper, we address this need by proposing a robust end-to-end argument parser that formulates AM as a dependency parsing (DP) problem. Unlike prior research in DP based argument mining approaches, we exploit the innate relationship between EDUs and ADUs in multi-task learning (MTL) framework and achieve competitive results. We further improve upon our results by utilizing appropriate contextual information and optimizing a novel objective function during training.

2 Related Work

Significant advancements have been made in computational model for AM in recent years. Stab and Gurevych (2014b) implemented a feature engineering based pipelined approach for performing all four sub-tasks of AM, on the Persuasive Essays (PE) corpus (Stab and Gurevych, 2014a), which was further improved by the Integer Linear Programming (ILP) based approach proposed by Persing and Ng (2016). Stab and Gurevych (2017) introduced a larger version of the PE corpus and implemented an ILP constrained pipelined approach for AM. Mirko et al. (2020) improved upon the pipelined approach for AM introduced by Nguyen and Litman (2018), and further implemented a novel graph construction process to create argument graphs. Recently, Bao et al. (2021) proposed a neural transition-based model for component classification and relationship detection, which incrementally builds an argumentation graph by generating a sequence of actions, and can handle both tree and non-tree argumentation structures.

Eger et al. (2017) formulated the tasks of AM as a token level DP, and achieved state-of-the-art performance on the PE dataset, using a neural dependency parser. Inspired by the success of incorporating biaffine classifiers for semantic DP (Dozat and Manning, 2016, 2018), Ye and Teufel (2021) further improved the DP based approach by using biaffine layers, and leveraged pre-trained BERT (Devlin et al., 2018) for richer argument representations. Instead of operating at a word level, Morio et al. (2020) experimented with proposition level AM and used a joint learning framework for jointly performing the tasks of component classification, relation detection and classification.

Considerable work has also been done in trying to establish relationships between ADUs and EDUs. Peldszus (2015); Peldszus and Stede (2016); Musi et al. (2018); Hewett et al. (2019) studied the mapping from discourse structure from Rhetorical Structure Theory (RST) to argumentation structures and showed that discourse relations from RST often correlate with argumentative relations.

3 Proposed Approach

Our work is inspired by the token level dependency parser proposed by Ye and Teufel (2021), and the proposition level parser proposed by Morio et al. (2020). However, unlike previous works, our formulation of dependency representation for arguments unifies all sub-tasks of AM under an EDU level framework and exploits the relationship between EDUs and ADUs. We factorize the sub-tasks of AM as different prediction tasks and train end-to-end in a multi-task learning (MTL) framework. We implement a hierarchical encoding scheme, which enables the use of a larger context, and train using a modified loss function for increasing performance.

3.1 Dependency Representation for Arguments

As illustrated in Figure 1, we structure arguments as a combination of EDUs and define types of relationships that could potentially hold between EDUs. We further enrich each EDU token with segment boundaries, that enable the re-construction of ADU from the EDUs. We list the properties of our dependency representation below:

- An EDU can either partially or fully overlap with an ADU and each token in an EDU is labeled as argumentative or non-argumentative, using the IO tagging scheme. For example in Figure 1, EDU 1 partially overlaps with ADU 1, as the token “because" is tagged as “O", whereas the EDUs 2 and 3 fully overlap with ADU 2, which is indicated by all the tokens in the EDUs labeled as “I".

- Each EDU can only belong to 1 of 4 classes $\in$ [major claim (MC), claim (C), premise (P), non-argument (NA)]. Consecutive EDUs which belong to the same class can be combined using *Append* relationship to yield an ADU. For example in Figure 1, EDU 2 and 3 can be combined using the *Append* relationship to construct ADU 2.

- Claim and premise EDUs can be related using “Support" (Sup) or “Attack" (Att) relationships, with the relationship originating from the last EDU of a claim to the last EDU of a premise. EDUs comprising premises can be related using the “Support" relationship, with the relationship originating from the last EDU of the supported premise to the last EDU of the supporting premise.

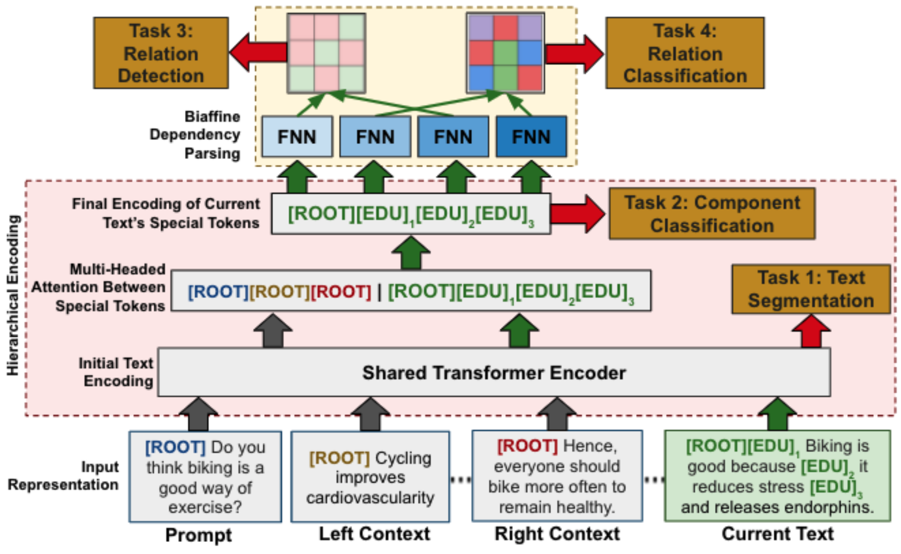

Task 3: Relation Detection
Task 4: Relation Classification
Biaffine Dependency Parsing
FNN
FNN
FNN
FNN
Hierarchical Encoding
Final Encoding of Current Text's Special Tokens
[ROOT][EDU]1[EDU]2[EDU]3
Task 2: Component Classification
Multi-Headed Attention Between Special Tokens
[ROOT][ROOT][ROOT] | [ROOT][EDU]1[EDU]2[EDU]3
Task 1: Text Segmentation
Initial Text Encoding
Shared Transformer Encoder
Input Representation
[ROOT] Do you think biking is a good way of exercise?
[ROOT] Cycling improves cardiovascularity
[ROOT] Hence, everyone should bike more often to remain healthy.
[ROOT][EDU]1 Biking is good because [EDU]2 it reduces stress [EDU]3 and releases endorphins.
Prompt
Left Context
Right Context
Current Text

Figure 2: End-to-End Model Architecture.

- A pseudo-token "ROOT" is added to the beginning of each argument, which represents the topic or gist of the entire argument. "ROOT" is always acting as a parent to the highest-level component(s).

- Each claim is parented by the "ROOT", and related using "For" (For) or "Against" (Agn) relationship, signifying the stance of the claim concerning the topic of discussion. In case of the presence of a MC, the "ROOT" parents the MC using a "default" (Def) relationship, which in turn parents the claims using "For" or "Against" relationships.

- An EDU can contain zero or more parents, and the relationships are acyclic.

In contrast to Ye and Teufel (2021), parsing arguments at an EDU level reduces complexity and simplifies all sub-tasks of AM, which we hypothesize should lead to better results. Similar to Morio et al. (2020), we implement DP only for the relationship detection and classification sub-tasks and further incorporate separate classifiers for text segmentation and component classification.

3.2 Multi-Task Learning (MTL) for AM

We train our argument parser in a MTL framework, where all the AM sub-tasks share a common encoding representation, followed by task-specific layers. Figure 2 illustrates our architecture in detail [1].

[1]EDU-AP codebase: https://github.com/sougata-ub/edu-ap.

3.2.1 Model Input Representation

We segment input text into EDUs using the BiLSTM-CRF based discourse segmenter by Wang et al. (2018) and add a special *[EDU]* token to the start of each span. The *[EDU]* token acts as a delimiter between EDU spans, and also represents the meaning of the corresponding EDU. Further, a *[ROOT]* token is added to the start of each input, which represents the meaning of the entire text.

3.2.2 Hierarchical Encoding

Depicted in Figure 2, we implement a hierarchical encoder, where we sequentially encode the current paragraph input and context tokens using a shared transformer encoder, and perform multi-headed attention (MHA) between the current input special tokens and the concatenated contextual *[ROOT]* tokens. Equations 1 to 4 defines the encoding process, where $\mathrm{E_{curr}}$ and $\mathrm{E_{ctx}}$ are the encoded representations of the current and context inputs $\mathrm{S_{curr}}$ and $\mathrm{S_{ctx}}$. The representations of the current turn and context special tokens $\mathrm{E}^{I}_{\mathrm{curr}}$ and $\mathrm{E}^{I}_{\mathrm{ctx}}$ are selected (using *Get*) from $\mathrm{E_{curr}}$ and $\mathrm{E_{ctx}}$ respectively. The final representation $\mathrm{E}^{\mathrm{MHA}}_{\mathrm{curr}}$ of the current turn's special tokens is obtained by sum pooling $\mathrm{E}^{I}_{\mathrm{curr}}$ and the MHA output followed by a dropout layer.

$$\mathrm{E_{curr}}{=}\mathrm{Encode}(\mathrm{S_{curr}});\, \mathrm{Get}(\mathrm{X}, idx){=}\mathrm{X}[idx, :] \quad (1)$$

$$\mathrm{E}^{I}_{\mathrm{curr}}{=}\mathrm{Get}(\mathrm{E_{curr}}, idx_{\mathrm{ROOT,EDU}}) \quad (2)$$

$$\mathrm{E}^{I}_{\mathrm{ctx}}{=}\mathrm{Get}(\mathrm{Encode}(\mathrm{S_{ctx}}), idx_{\mathrm{ROOT}}) \quad (3)$$

$$\mathrm{E}^{\mathrm{MHA}}_{\mathrm{curr}}{=}\mathrm{E}^{I}_{\mathrm{curr}}{+}\mathrm{Dropout}(\mathrm{MHA}(\mathrm{E}^{I}_{\mathrm{curr}}, \mathrm{E}^{I}_{\mathrm{ctx}})) \quad (4)$$

Such a formulation not only encourages the special tokens to better encode its representative span but also ameliorates the length bottleneck (Joshi et al., 2020) in transformer architectures by reducing the sequence length. Thus, enabling the use of a larger context compared to token level parsing.

3.2.3 Task Specific Prediction

Post encoding, we incorporate task-specific layers to perform the final prediction for each task. Depicted in Equations 5 and 6, we use single-layered feed-forward neural networks (FNNs) as the final layer for both text segmentation and component classification. For text segmentation, we use the initial token level encoding ($\mathrm{E_{curr}}$) of the current text as input, whereas for component classification, the final encoding of the current text *[EDU]* tokens ($\mathrm{E_{curr}^{MHA}}$) are used as inputs.

$$sc^{span}=\mathrm{FNN}(\mathrm{E_{curr}});\ sc^{typ}=\mathrm{FNN}(\mathrm{E_{curr}^{MHA}}) \quad (5)$$

$$y'^{span}=\{sc^{span} \geq 0\};\ y'^{typ}=\mathrm{argmax}\ sc^{typ} \quad (6)$$

Biaffine classifiers are generalizations of linear classifiers, which include multiplicative interactions between two vectors. Since relation detection and relation classification require performing inference between argument pairs, we implement biaffine dependency parsing (DP) for both the tasks. Using FNNs, the current text *[ROOT]* and *[EDU]* encodings $\mathrm{E_{curr}^{MHA}}$ are split into two parts with reduced hidden size–a head (parent) and a dependent (child) representation, which in turn are passed through a biaffine classifier (Biaf) for predicting edges and labels between EDUs. Equations 7 to 11 details our biaffine DP formulation, where $\mathrm{H^{e\_p}}$ and $\mathrm{H^{e\_c}}$ denotes the parent and child representations for relation detection, and $\mathrm{H^{l\_p}}$ and $\mathrm{H^{l\_c}}$ denotes the parent and child representations for relation classification. sc^e and sc^l contains the output logits from the biaffine layers, where $sc^e_{i,j}$ and $sc^l_{i,j}$ denotes the logits between the i^{th} and j^{th} EDU for relation detection and classification respectively.

$$\mathrm{Biaf}(x,y)=x^T\mathrm{U}y+\mathrm{W}(x \oplus y)+b \quad (7)$$

$$\mathrm{H^{e\_p}}=\mathrm{FNN}(\mathrm{E_{curr}^{MHA}});\ \mathrm{H^{e\_c}}=\mathrm{FNN}(\mathrm{E_{curr}^{MHA}}) \quad (8)$$

$$\mathrm{H^{l\_p}}=\mathrm{FNN}(\mathrm{E_{curr}^{MHA}});\ \mathrm{H^{l\_c}}=\mathrm{FNN}(\mathrm{E_{curr}^{MHA}}) \quad (9)$$

$$sc^e=\mathrm{Biaf}(\mathrm{H^{e\_p}},\mathrm{H^{e\_c}});\ sc^l=\mathrm{Biaf}(\mathrm{H^{l\_p}},\mathrm{H^{l\_c}}) \quad (10)$$

$$y'^e_{i,j}=\{sc^e_{i,j} \geq 0\};\ y'^l_{i,j}=\mathrm{argmax}\ sc^l_{i,j} \quad (11)$$

3.2.4 Modified Objective Function

Depicted in Equation 16, we train the model end-to-end by minimizing the aggregated interpolated loss across all four sub-tasks, with an interpolation factor λ. The sub-tasks of text segmentation, component classification and relation classification are trained by minimizing the cross entropy (CE) losses $\mathcal{L}^{span}$, $\mathcal{L}^{typ}_i$, $\mathcal{L}^l_{i,j}$ respectively in Equations 14 and 15, whereas relation prediction is trained by minimizing the binary cross entropy (BCE) loss $\mathcal{L}^e_{i,j}$ (Equation 13).

We further add an extra penalty term δ with an interpolation factor of β to the BCE loss in Equation 13, to increase the recall of predicting relationships between EDUs. Exploiting the symmetry of the final score matrix (logits) in biaffine classifiers, as depicted in Equations 12 and 13, the penalty term for the relationship from the i^{th} to j^{th} EDU is set to be dependent on the logit of its reverse: j^{th} to i^{th}. This results in the loss function being penalized most in case a relationship and its conjugate reverse are both predicted to be present or absent, and least if either is predicted to exist. We hypothesize that such a penalty should increase the relation detection recall, while minimally impacting the precision.

$$\delta(y,\tilde{y})=y\log(1\text{-}\sigma(\tilde{y})) + (1\text{-}y)\log\sigma(\tilde{y}) \quad (12)$$

$$\mathcal{L}^e_{i,j}=\beta\ \mathrm{BCE}(\mathrm{y^e_{i,j}},\mathrm{sc^e_{i,j}})\text{-}(1\text{-}\beta)\delta(y^e_{i,j},sc^e_{j,i}) \quad (13)$$

$$\mathcal{L}^l_{i,j}=\mathrm{CE}(y^l_{i,j},sc^l_{i,j});\ \mathcal{L}^{typ}_i=\mathrm{CE}(y^{typ}_i,sc^{typ}_i) \quad (14)$$

$$\mathcal{L}^{span}=\mathrm{CE}(y^{span},sc^{span}) \quad (15)$$

$$\mathcal{L}=\lambda\mathcal{L}^e+(1\text{ - }\lambda)(\mathcal{L}^l+\mathcal{L}^{span}+\mathcal{L}^{typ}) \quad (16)$$

3.2.5 Post Processing and Graph Construction

During inference, we perform a few post-processing steps to constrain the model output. For relationship prediction, we discard self references and cyclic relationships, and further restrict the argument structures to conform to the ones defined by Stab and Gurevych (2017), i.e premise→premise, premise→claim, claim→major claim/claim→ROOT and major claim→ROOT.

For generating an argument graph, we extract ADUs by concatenating consecutive argumentative EDUs that are predicted to be connected by an “Append" relationship and remove non-argumentative tokens, using the text segmentation prediction. To yield contiguous arguments and prevent unnatural segmentation, we ensure the label of each token within an appended ADU confirms with its neighbours, and re-label to the majority class of its neighbours if needed. Next, we label each ADU by assigning the majority label of the constituent EDUs predicted by the component classifier. Finally, a graph is formed by connecting the labeled ADUs with the relationships predicted by the relation classifier, only if the relation detector predicts its existence.

4 Experiments

4.1 Dataset

We use the benchmark persuasive essays (PE) dataset by Stab and Gurevych (2017), for all our experiments. This dataset comprises 402 persuasive essays: 322 for training and 80 for testing, randomly selected from an online forum. Barring non-arguments, there are three kinds of argumentative components in the dataset, along with four types of relationships that can hold between the components: *(i) Major claim*: the main claim by an author, which informs their stance. *(ii) Claim*: a statement that is either *For* or *Against* one or more major claims. *(iii) Premise*: a statement that provides evidence to a claim or another premise by a *Support* or *Attack* relationship. Please refer Stab and Gurevych (2017) for a detailed dataset statistics.

4.2 Experiment Setup

We use Roberta (base) (Liu et al., 2019) as the base encoder, and increase its embedding layer to accommodate the special tokens. Two layers comprising four attention heads are used for MHA, where the MHA result in each layer is sum pooled with the residual output while applying dropout with 0.1 probability to the MHA result. The hidden size of the FNNs in the biaffine layer is set to 600. An interpolation factor λ of 0.95 is used for aggregating the losses, and the factor β in the modified BCE loss is set to 0.85. All models are trained with a learning rate of 1e-5 for 15 epochs and optimised using AdamW (Loshchilov and Hutter, 2017), with early stopping if the validation loss doesn't reduce for 2 epochs. We repeat each experiment five times and report the average across all runs.

4.3 Competing Model

We recreate the state-of-the-art BiPAM parser by Ye and Teufel (2021) as an external baseline and compare it against our proposed method. BiPAM implements token level dependency parsing for end-to-end argument mining and had achieved significant improvements over LSTM-Parser and LSTM-ER reported in Eger et al. (2017).

4.4 Evaluation Metrics

All tasks are evaluated using the F1 score. Similar to Persing and Ng (2016), approximate and exact overlap is computed between the golden and predicted ADUs, where a predicted ADU is classified as approximate overlap if at least 50% of its tokens match with the golden ADU, and is classified as exact overlap if all the tokens match with the golden ADU. Each task is evaluated for both the approximate and exact overlapping ADU spans.

4.5 Results and Analysis

We share the experimental results for the sub-tasks of text segmentation and component classification in Table 1, and the sub-tasks of relation detection and relation classification in Table 2. In each table, we calculate and report the F1 score for both approximate and exact overlapping ADUs (approx/exact). We treat EDU-AP–our non-contextual implementation without the δ loss penalty as the internal baseline, and underline the best performing model for each task, in comparison to this baseline. We also compare the results obtained from the BiPAM parser and highlight the best performing result in comparison to this baseline in bold.

As indicated by the top section of Table 1, using a mixture of EDUs and tokens, our baseline EDU-AP outperforms token level BiPAM for text segmentation by a significant margin (61.8/53.2 compared to 38.8/25.6). We reason that operating solely with tokens, BiPAM increases the chances of locally erroneous predictions, yielding discontinuities in ADU spans. Whereas EDU-AP minimizes this by globally identifying EDUs which should be combined as an ADU using the "Append" relationship, and further locally eliminating non-argumentative tokens from each EDU. This is further corroborated by the fact that the average ratio between predicted ADU spans and golden ADU spans per paragraph is 1.1 for the EDU-AP, in comparison to an average ratio of 2.2 in the BiPAM parser, signifying a greater number of short spans predicted by BiPAM.

For component classification (Table 1), EDU-AP outperforms BiPAM across all classes (Major Claim: MC, Claim: C, Premise: P and Non Argument: NA) for the approximately matched ADU spans. However, for the exact matching spans, BiPAM mostly performs better than EDU-AP.

For both the relation detection and classification tasks in the top section of Table 2, EDU-AP largely

	ADU Span	Component Classification			
Model	**F1**	**MC-F1**	**C-F1**	**P-F1**	**NA-F1**
BiPAM †	38.8 / 25.6	81.9 / 87.8	67.8 / **77.0**	88.8 / 88.0	92.5 / **98.3**
EDU-AP *	**61.8** / 53.2	86.2 / 86.8	68.9 / 70.8	90.3 / 90.7	96.4 / 97.6
EDU-AP + prompt	61.4 / 52.5	84.9 / 85.2	69.2 / 71.7	90.5 / 91.1	96.6 / 97.7
EDU-AP + left	61.5 / 51.9	84.2 / 83.9	64.8 / 67.4	89.9 / 90.1	96.2 / 97.6
EDU-AP + all	60.5 / 50.8	83.6 / 84.1	68.2 / 70.2	90.5 / 90.8	95.9 / 97.7
EDU-AP + δ	61.4 / **53.5**	85.7 / 87.3	73.5 / 75.1	91.2 / 91.4	97.2 / 98.1
EDU-AP + δ + prompt	61.2 / 53.2	**87.8** / 88.1	**74.2** / 76.1	**91.9** / **92.2**	**97.3** / 97.9
EDU-AP + δ + left	60.0 / 51.5	87.0 / **88.4**	71.2 / 73.0	91.5 / 91.6	96.6 / 97.8
EDU-AP + δ + all	59.9 / 51.6	86.3 / 86.6	71.9 / 73.7	91.3 / 91.6	96.0 / 97.4

Table 1: Results for the sub-tasks Text Segmentation and Component Classification (**M**ajor **C**laim, **C**laim, **P**remise, **N**on **A**rgument), for both approximate and exact overlapping spans (approx/exact). †and * denotes external and internal baselines respectively.

	Relation Detection	Relation Classification				
Model	**F1**	**Agn-F1**	**Att-F1**	**Def-F1**	**For-F1**	**Sup-F1**
BiPAM †	37.1 / 13.2	16.1 / 5.0	0.0 / 0.0	61.1 / 30.9	51.3 / 24.4	26.1 / 5.7
EDU-AP *	57.0 / 47.3	56.5 / 46.9	31.9 / 12.4	85.5 / 72.9	74.1 / 65.6	57.7 / 47.5
EDU-AP + prompt	62.0 / 51.6	58.5 / 47.7	35.3 / 14.1	85.1 / 71.2	81.9 / 73.2	68.8 / 57.3
EDU-AP + left	57.2 / 46.8	55.9 / 48.3	29.8 / 14.1	86.5 / 71.0	75.4 / 66.9	59.6 / 48.2
EDU-AP + all	57.8 / 47.4	56.4 / 47.2	31.1 / 15.6	86.5 / 71.1	75.9 / 66.9	59.5 / 48.8
EDU-AP + δ	62.6 / 53.8	64.6 / **57.0**	38.8 / 18.7	86.3 / **77.8**	80.3 / 73.2	69.1 / 58.5
EDU-AP + δ + prompt	**64.9** / **55.0**	64.4 / 56.7	**41.1** / **22.6**	**88.4** / 77.0	**82.5** / **74.1**	**74.3** / **62.9**
EDU-AP + δ + left	62.3 / 52.2	59.3 / 50.9	39.0 / 17.3	86.1 / 75.9	77.7 / 69.9	66.8 / 54.8
EDU-AP + δ + all	62.8 / 53.0	64.4 / 57.0	37.7 / 18.1	87.8 / 76.7	81.1 / 72.5	69.3 / 58.0

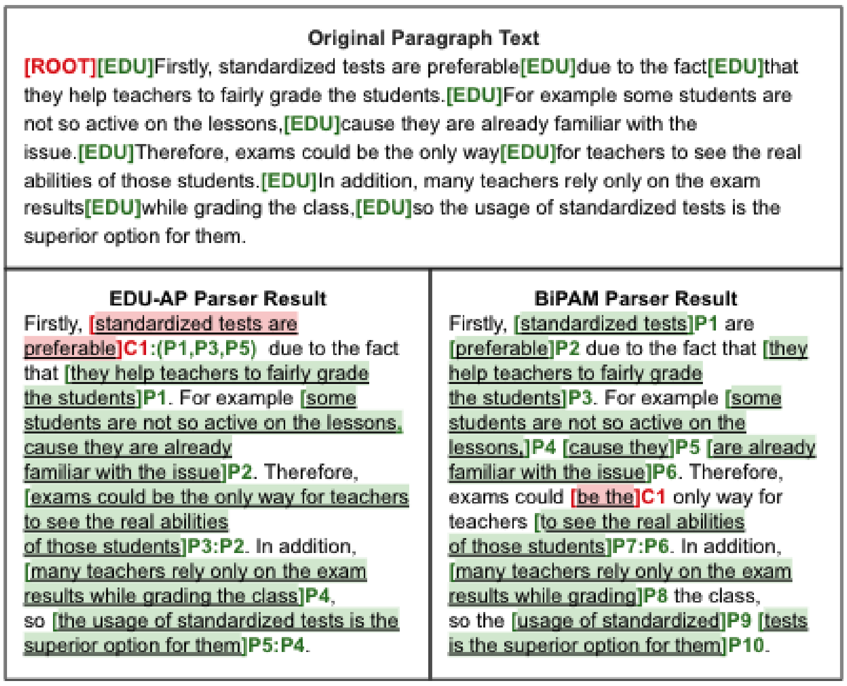
Table 2: Results for the sub-tasks Relation Detection and Relation Classification (**Ag**ai**n**st, **Att**ack, **Def**ault, **For**, **Sup**port), for both approximate and exact overlapping spans (approx/exact). †and * denotes external and internal baselines respectively.

outperforms BiPAM. We reason that since both the tasks demand inference over pairs of argumentative sentences, using our formulation of operating at an EDU level and using representative *[EDU]* tokens better represents and encodes arguments, thus providing better context during scoring, compared to operating at the individual token level. Further, unlike BiPAM, incorporating task-specific layers encourages learning task-specific parameters, which enhances the model's performance.

4.6 Ablation Study

We further perform an ablation study, to determine the effect of adding the δ penalty and utilizing context in all sub-tasks. The bottom section in both Tables 1 and 2 includes the ablation results. We experiment with combinations of adding an essay's prompt as context (+prompt), the prompt along with all the past paragraphs (+left), the prompt along with all other paragraphs (+all), and the loss penalty (+δ).

Overall, we observe that although baseline EDU-AP performs better than BiPAM, incorporating the δ penalty increases the model's efficacy for most sub-tasks, which is further boosted by adding the essay's prompt (+prompt) as context. Most significant improvements are observed for relation detection and classification (Table 2 bottom section), which is intuitively justified, as establishing relationships (like *For/Against*) not only require knowledge and understanding of the main theme of discussion, but also cognizance of the established stance towards the topic from prior paragraphs. Table 3 further illustrates the impact of the δ penalty on the precision and recall scores for relation detection. As previously hypothesized, we observe



Original Paragraph Text

[ROOT][EDU]Firstly, standardized tests are preferable[EDU]due to the fact[EDU]that they help teachers to fairly grade the students.[EDU]For example some students are not so active on the lessons,[EDU]cause they are already familiar with the issue.[EDU]Therefore, exams could be the only way[EDU]for teachers to see the real abilities of those students.[EDU]In addition, many teachers rely only on the exam results[EDU]while grading the class,[EDU]so the usage of standardized tests is the superior option for them.

EDU-AP Parser Result

Firstly, [standardized tests are preferable]C1:(P1,P3,P5) due to the fact that [they help teachers to fairly grade the students]P1. For example [some students are not so active on the lessons, cause they are already familiar with the issue]P2. Therefore, [exams could be the only way for teachers to see the real abilities of those students]P3:P2. In addition, [many teachers rely only on the exam results while grading the class]P4, so [the usage of standardized tests is the superior option for them]P5:P4.

BiPAM Parser Result

Firstly, [standardized tests]P1 are [preferable]P2 due to the fact that [they help teachers to fairly grade the students]P3. For example [some students are not so active on the lessons,]P4 [cause they]P5 [are already familiar with the issue]P6. Therefore, exams could [be the]C1 only way for teachers [to see the real abilities of those students]P7:P6. In addition, [many teachers rely only on the exam results while grading]P8 the class, so the [usage of standardized]P9 [tests is the superior option for them]P10.

Figure 3: Comparison of a parsed paragraph from test set using the EDU-AP (left) and BiPAM (right) parsers.

that incorporating the δ penalty almost always improves recall, while also boosting the precision in some cases.

Model	F1	Precision	Recall
BiPAM †	37.1 / 13.2	32.3 / 13.9	45.4 / 13.1
EDU-AP *	57.0 / 47.3	71.6 / 66.1	48.1 / 37.6
+ prompt	62.0 / 51.6	68.0 / 62.1	57.1 / 44.2
+ left	57.2 / 46.8	68.6 / 62.4	49.4 / 37.7
+ all	57.8 / 47.4	70.2 / 64.2	50.2 / 38.5
+ δ	62.6 / 53.8	69.2 / 64.5	57.5 / 46.4
+ δ + prompt	**64.9 / 55.0**	67.8 / 62.5	**62.3 / 49.2**
+ δ + left	62.3 / 52.2	**72.8 / 67.7**	55.0 / 42.9
+ δ + all	62.8 / 53.0	69.0 / 63.7	58.1 / 45.7

Table 3: Comparison of F1, Precision and Recall for the Relation Detection subtask, for both approximate and exact overlapping spans (approx/exact). † and * denotes external and internal baselines respectively.

We also observe that text segmentation largely remains unaffected by the addition of context and δ penalty (Table 1 bottom section), which is justified by the nature of the task, which does not depend much on external context, and relies more on linguistic features.

The nature of argumentation is such that the label of the components can be ascertained with a fair probability, from the relationships that exist between components. For example, as described in sub-section 4.1, only a claim can be the child node in a *For/Against* relationship, and premises can only be a part of *Support/Attack* relationships. Although the δ penalty is not directly applied to component classification, we still observe an increase in performance for classifying components, with the addition of the δ penalty and context (Table 1 bottom section). We attribute this to our multi-task learning framework, which enables learning joint representations that benefit all sub-tasks.

It is also interesting to note that for all the sub-tasks, adding more context (+left, +all) does not always yield superior results, whereas just adding the prompt (+prompt) of the essay yields better results. We attribute this to the fairly small size of the corpus used in the experiments, which does not provide many data points for learning complex interactions from context.

4.7 Discussion

Our results indicate that parsing text at a combination of EDU and token level yields better results, compared to bare token level DP, and can be further improved by appropriately penalizing the loss function, and incorporating contextual information. Figure 3 illustrates and compares a parsed example of a paragraph from the test set, using both the EDU-AP and BiPAM parsers. We underline and enclose the predicted argumentative spans by the models in square brackets, and assign a unique identifier to each component (C1, C2, P1, etc.). Predicted claims are highlighted in red, and their unique identifier starts with ‘C’, whereas premises are highlighted in green, with their identifier starting with ‘P’. Predicted relationships between components are separated using a colon. Example P5:P4 signifying support relationship from P4 to P5, or C1:(P1, P3, P5) signifying support relationship from P1, P3 and P5 to C1.

We observe that EDU-AP can correctly identify ADU spans by merging EDUs using the “Append" relationship and further eliminating non-argumentative tokens. BiPAM on the other hand yields more fragmented and discontinuous spans. The average ratio of predicted and golden ADU spans per paragraph in the test set is 1.1 for the EDU-AP in comparison to 2.2 for BiPAM parser, signifying a greater number of shorter spans predicted by BiPAM. For example, the span C1 in EDU-AP parsed output (which matches with the golden span) is split into two spans: P1 and P2 by BiPAM. We further observe that EDU-AP is not only able to correctly label the identified ADUs as claim and premise but also able to correctly predict support relationships between the ADUs. Compared to that, BiPAM is not able to correctly identify the claim of the paragraph and fails to predict any relationships between the arguments.

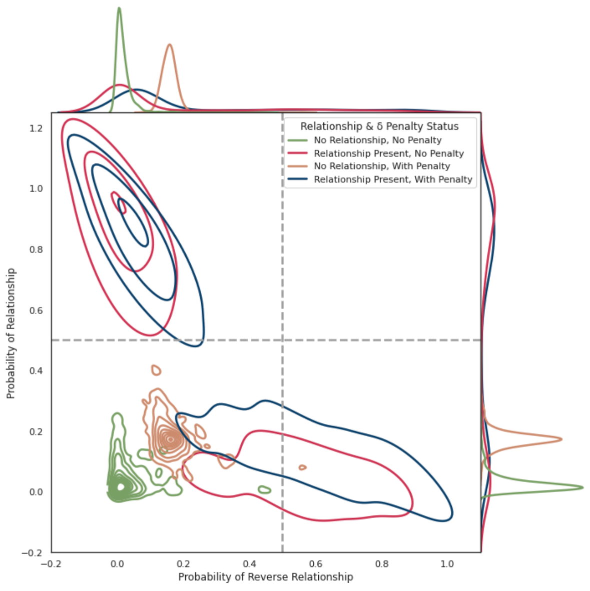


Figure 4: KDE plots comparing the effect of δ penalty on the distribution of relationship probability and its reverse for relationship detection. The dotted line denotes the probability threshold used for the experiments.

To understand the effect of the δ penalty on relation detection, we combine results from all experiments and plot the kernel density of probabilities of predicted relationships and its conjugate reverse relationship in Figure 4. We observe that as expected, overall the model assigns lower probabilities when no relationship exists between a pair of argument components and asymmetrically higher probabilities when a relationship exists, signifying a unidirectional relationship. Adding the δ penalty has the effect of shifting the probability distributions towards more symmetry (i.e for pairs of components, the difference of predicted probability for both directions is reduced), resulting in a recall seeking behaviour.

Although EDU-AP outperforms all baselines, it still fails to attain the human upper bound performance measured by Stab and Gurevych (2017) on the PE corpus. Further, trained only on monological essay data, EDU-AP can’t be used for parsing other forms of discourse like dialogue, which we seek to address in our next research steps.

5 Conclusion

In this paper, we present EDU-AP, an end-to-end dependency parsing based argument parser for parsing arguments from molonogical text. Exploiting the innate relationship between EDUs and ADUs, along with the appropriate use of context, and a hierarchical encoding scheme, EDU-AP is trained end-to-end in a multi-task learning setting by minimizing a novel loss function. EDU-AP’s efficacy is demonstrated by its superior experimental and ablation results, in comparison to strong internal and external baselines. We believe, with minor adjustments EDU-AP can be purposed for parsing arguments from dialogues.

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, Online. Association for Computational Linguistics.

Lisa Andreevna Chalaguine and Anthony Hunter. 2020. A persuasive chatbot using a crowd-sourced argument graph and concerns. In *COMMA*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Timothy Dozat and Christopher D. Manning. 2016.

Deep biaffine attention for neural dependency parsing.

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, Melbourne, Australia. Association for Computational Linguistics.

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, Vancouver, Canada. Association for Computational Linguistics.

Freya Hewett, Roshan Prakash Rane, Nina Harlacher, and Manfred Stede. 2019. The utility of discourse parsing features for predicting argumentation structure. In *Proceedings of the 6th Workshop on Argument Mining*, pages 98–103, Florence, Italy. Association for Computational Linguistics.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. 2020. Spanbert: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77.

John Lawrence and Chris Reed. 2019. Argument mining: A survey. *Computational Linguistics*, 45(4):765–818.

Marco Lippi and Paolo Torroni. 2016. Argumentation mining: State of the art and emerging trends. *ACM Trans. Internet Technol.*, 16(2).

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization.

LENZ Mirko, Premtim Sahitaj, Sean Kallenberg, Christopher Coors, Lorik Dumani, Ralf Schenkel, and Ralph Bergmann. 2020. Towards an argument mining pipeline transforming texts to argument graphs. *Computational Models of Argument: Proceedings of COMMA 2020*, 326:263.

Raquel Mochales and Marie-Francine Moens. 2011. Argumentation mining. *Artificial Intelligence and Law*, 19(1):1–22.

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, Online. Association for Computational Linguistics.

Elena Musi, Manfred Stede, Leonard Kriese, Smaranda Muresan, and Andrea Rocci. 2018. A multi-layer annotated corpus of argumentative text: From argument schemes to discourse relations. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Huy V. Nguyen and Diane J. Litman. 2018. Argument mining for improving the automated scoring of persuasive essays. In *AAAI*.

Andreas Peldszus. 2015. An annotated corpus of argumentative microtexts.

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. 2016. Rhetorical structure and argumentation structure in monologue text. In *Proceedings of the Third Workshop on Argument Mining (ArgMining2016)*, pages 103–112, Berlin, Germany. Association for Computational Linguistics.

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, San Diego, California. Association for Computational Linguistics.

Noam Slonim, Yonatan Bilu, Carlos Alzate, Roy Bar-Haim, Ben Bogin, Francesca Bonin, Leshem Choshen, Edo Cohen-Karlik, Lena Dankin, Lilach Edelstein, et al. 2021. An autonomous debating system. *Nature*, 591(7850):379–384.

Christian Stab and Iryna Gurevych. 2014a. Annotating argument components and relations in persuasive essays. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 1501–1510, Dublin, Ireland. Dublin City University and Association for Computational Linguistics.

Christian Stab and Iryna Gurevych. 2014b. Identifying argumentative discourse structures in persuasive essays. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 46–56.

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

Yizhong Wang, Sujian Li, and Jingfeng Yang. 2018. Toward fast and accurate neural discourse segmentation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 962–967, Brussels, Belgium. 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, Online. Association for Computational Linguistics.