Sentence-Level Discourse Parsing as Text-to-Text Generation

Anonymous ACL submission

Abstract

Previous studies have made great advances in RST discourse parsing through neural frameworks or efficient features, but they split the parsing process into two subtasks and heavily depended on gold segmentation. In this paper, we introduce an end-to-end method for sentence-level RST discourse parsing via transforming it into a text-to-text generation task. Our method unifies the traditional two-stage parsing and generates the parsing tree directly from the input text without requiring a complicated model. Moreover, the EDU segmentation can be simultaneously generated and extracted from the parsing tree. Experimental results on the RST Discourse Treebank demonstrate that our proposed method outperforms existing methods in both tasks of sentence-level RST parsing and discourse segmentation. Considering the lack of annotated data in RST parsing, we also create high-quality augmented data based on several filtering strategies, which further improves the performance.

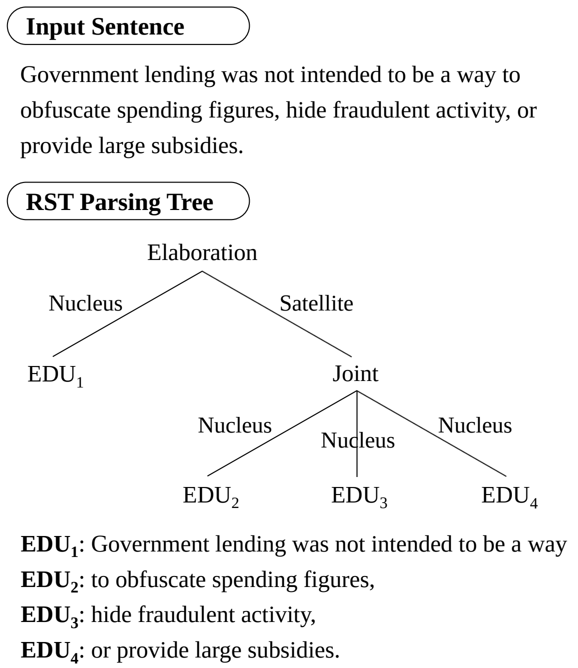


Figure 1: An example from RST Discourse TreeBank.

1 Introduction

Discourse parsing involves determining the structure of elementary units forming a discourse and how they are connected with each other. In a coherent text, units are often organized logically and semantically with certain relationships. Early studies have demonstrated that discourse parsing can benefit various downstream NLP tasks, including sentiment analysis (Polanyi and van den Berg, 2011; Bhatia et al., 2015), summarization (Louis et al., 2010; Gerani et al., 2014), question answering (Jansen et al., 2014) and machine translation evaluation (Joty et al., 2017).

RST parsing based on Rhetorical Structure Theory (Mann and Thompson, 1987), is one of the most common and influential parsing methods in discourse analysis. According to RST, a text is first segmented into several clause-like units as leaves of the corresponding parsing tree, called elementary discourse units (EDUs). Through certain rhetorical relations among adjacent spans, such as Elaboration and Joint, underlying EDUs or larger text spans are recursively linked and merged to form their parent nodes, representing the concatenation of them. Finally, a hierarchical tree structure is constructed. Besides rhetorical relations, sibling nodes in the parsing tree contain a kind of nucleus-satellite relations to show who is more central or equal to the discourse structure. Figure 1 shows an RST parsing tree for a sentence from the RST Discourse TreeBank (Carlson and Marcu, 2001), which is the most common discourse corpus.

In the past, various approaches have been proposed for both document-level and sentence-level RST parsing, which can be mainly divided into bottom-up and top-down methods. Earlier work like transition-based approaches utilized the representation learned through manually-designed features or neural networks to build shift-reduce parsers (Ji and Eisenstein, 2014; Yu et al., 2018). The whole parsing tree is gradually built in a sequence of actions, including shift and reduce. Then, benefiting from the development of neural networks, top-down approaches (Lin et al., 2019; Liu et al., 2019; Zhang et al., 2020) made use of the pointer network (Vinyals et al., 2015) to segment text into shorter units recursively until no more units can be generated.

Although many advances have been made in RST parsing, the real performance of existing methods may be far from satisfactory. Most studies before followed the traditional settings to split the parsing process into two stages, namely segmenting EDUs and building parsing trees. They employed their models only on the second stage and treated the gold EDU segmentation as a requisite, which is, however, infeasible in real application scenarios. The segmenter trained in the first stage can generate automatic segmentation as a substitute, but the performance of those parsing methods would drop a lot accordingly. This may be caused by errors in segmenters transmitting to the parsing stage. Moreover, previous methods relied on additional features or complicated frameworks for different parts of parsing like relation label prediction, which did not take full advantage of knowledge in the task.

In this paper, we focus on sentence-level RST parsing and introduce a simple end-to-end method which can generate the target parsing tree directly from the corresponding text. It is beneficial since sentence-level discourse analysis has relatively high accuracy and can be applied to many NLP tasks like sentence compression (Soricut and Marcu, 2003). Moreover, sentence-level parsing is essential and serves as a basic step in some document-level parsers (Wang et al., 2017; Kobayashi et al., 2020). Therefore, the improvement of sentence-level parsing may promote further progress in discourse parsing.

Our proposed method converts RST parsing into a text-to-text generation task by reformulating the parsing tree into a natural language sequence. The information contained in text content, hierarchical structures, and relation labels in the parsing tree can be integrated and learned together by the generation model. Experimental results demonstrate that our method outperforms existing approaches without using gold segmentation. In addition, our method can generate the EDU segmentation simultaneously during parsing, which has even better performance than other segmenters specifically trained on this task. In view of the lack of annotated data in RST parsing, we also attempt to generate high-quality augmented data to obtain extra enhancement.

Our primary contributions are as follows: (1) we propose a simple but effective end-to-end approach to sentence-level RST parsing without using gold segmentation and additional auxiliary information; (2) our method generates the parsing tree with the EDU segmentation simultaneously and outperforms existing models on both tasks; (3) we attempt to generate augmented data according to certain strategies to further improve the performance. The code will be released to the community.

2 Related Work

Discourse parsing describes the hierarchical tree structure of a text and can be used in quality evaluation like coherence and other downstream applications. In the past, various approaches on both sentence-level and document-level RST parsing have been proposed, mainly divided into two classes: top-down and bottom-up paradigms.

In earlier studies, bottom-up methods have been first purposed since various kinds of hand-engineered features were mainstream tools and more suitable to represent local information. Soricut and Marcu (2003) first proposed a bottom-up CKY-like approach with syntactic and lexical features for sentence-level parsing. Models with CKY-like algorithms (Hernault et al., 2010; Joty et al., 2013; Feng and Hirst, 2014; Li et al., 2014) utilized diverse features to learn the scores for different subtrees and searched all possible parsing trees to find the most likely one for a text. Although these methods achieved high accuracy, they suffered from slow parsing speed.

Another common bottom-up method is the transition-based parser, which generates the RST parsing tree during a sequence of shift and reduce action decisions. Ji and Eisenstein (2014) introduced a neural shift-reduce parser with representation learning methods. Wang et al. (2017) proposed a two-stage parser based on SVMs with plenty of features. Then Yu et al. (2018) trained a transition-based parser with implicit syntactic features from dependency parsing and achieved great success. Although transition-based methods can benefit from low time complexity, they only unitize the local information for each decision and may not achieve the best result in the long run.

Thanks to the recent advancement of neural methods, it is possible to represent the text effectively in a global view, which promoted top-down parsers. Lin et al. (2019) first presented a seq2seq model for sentence-level RST parsing based on pointer networks (Vinyals et al., 2015) and Liu et al. (2019) improved it with hierarchical structure. Then Zhang et al. (2020) extended their methods to document-level RST parsing. Kobayashi et al. (2020) constructed subtrees for three granularity levels of text and merged them together.

Despite the better performance of top-down models, most of them still utilized gold EDU segmentation as a necessity and dropped a lot in performance when using automatic segmenters. However, it is more practical that the parsing tree should be constructed directly from the input text. And the two-stage process may lead to error accumulation from segmenting to parsing. Nguyen et al. (2021) introduced an end-to-end parsing model, but it relied on different frameworks for structure and relation label prediction and improved the performance with the help of artificial sentence guidance. In addition, we find contemporaneous work of Zhang et al. (2021) just before our submission. They introduce a complicated system with rerankers and we follow ACL's policy and do not make comparisons with this work. Our end-to-end approach, on the other hand, transforms RST parsing into a text generation task, eliminating the need for additional knowledge and specific frameworks.

3 Our Method

Over the past year, a new paradigm in NLP emerged based on powerful pretrained language models and brought remarkable improvement on many tasks. Instead of adapting pretrained models to different downstream tasks through specific network layers and objective engineering, now downstream tasks are reformulated close to the tasks used during pretraining (Liu et al., 2021). Many studies have proved that knowledge contained in pertrained models can be used directly to deal with text classification or generation. However, it still remains a significant challenge for more complex data structures, such as the tree structure in RST parsing.

Motivated by the idea above, we propose a method to reformulate the parsing tree into the form of a linear sequence so as to utilize existing seq2seq models. We show that our new text-to-text task can make great use of the latent knowledge in pretrained models like T5, without additional features or neural frameworks. Although the target output is not the parsing tree, it can be restored and evaluated through a series of post-processes, resulting in more accurate predictions.

3.1 Binarization

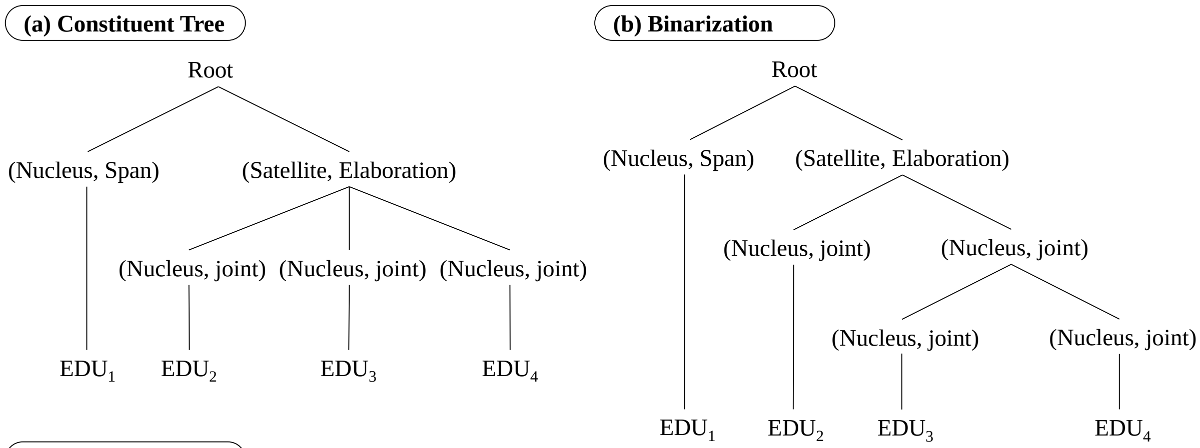
In the original RST Discourse TreeBank, RST parsing trees are stored as a set of text spans together with their relation labels. To a mononuclear relation, the span of the satellite is assigned a certain rhetorical relation, and that of the nucleus is assigned the label Span. Multinuclear relations hold among two or more spans of the nucleus, which are assigned the same rhetorical relations. Marcu (2000) first formally encoded the RST parsing tree in the form of a constituent tree, as shown in Figure 2(a), which was followed and used by the majority of subsequent parsing methods.

On the other hand, there are some n-ary trees in the corpus because multinuclear relations can link more than two spans, namely nodes in the parsing tree. The standard process is to turn them into their right-heavy binarized versions. Both of the processes above aim to make parsing trees more regular and suitable for training and evaluation. Since they also help to linearize the parsing tree in our method, we perform the same steps before the linearization. The relation labels are all the same for the leaves of minimum n-ary subtrees, so new intermediate nodes added during binarization just need to be assigned the same labels. The binary constituent tree in Figure 2(b) is transformed from the examples in Figure 1 and Figure 2(a).

3.2 Linearization

Based on the priority level contained in brackets, we attempt to represent hierarchical architecture by nesting several pairs of brackets. The linearization is carried out from the bottom up according to postorder traversal. We replace each leaf that represents a single EDU with a sequence comprised of a left bracket, text content, a right bracket, and its nuclearity and rhetorical relation labels. Blank characters are added to each interval between different elements.

As for intermediate nodes, we perform the same process except that the concatenation of new representations of two child nodes serves as the text content. Since the root does not contain any la-



(c) Linearization

[[Government lending was not intended to be a way] Nucleus span [[to obfuscate spending figures,] Nucleus joint [[hide fraudulent activity,] Nucleus joint [or provide large subsidies.] Nucleus joint] Nucleus joint] Satellite elaboration]

Figure 2: The process of reformulation for the RST parsing tree from Figure 1 according to our method.

bels, it simply merges two child nodes with a pair of outermost parentheses. The postorder traversal ensures that intermediate nodes will be processed after their child nodes are updated, and the root is the last one to be considered, resulting in the final linear sequence of the parsing tree.

Benefiting from binarization, the format of reformulated sequences is unified, with each pair of inner brackets followed by two relation labels, which can be better understood by pretrained language models. Considering that Paolini et al. (2021) proved and encouraged the use of the entire input to promote the performance, our linear sequence is designed to contain a complete copy of the corresponding input text. Besides, we use square brackets in linearization to avoid confusion since the input text itself may contain parentheses. The target linear sequence of the RST parsing tree in Figure 2(b) is shown in Figure 2(c).

3.3 Seq2seq Training

Since the input and new output of the task are both sequences, RST parsing can thus be trained or fine-tuned on any generation model as a text-to-text generation task. Pretrained seq2seq models like T5 (Raffel et al., 2020) are able to transfer the related latent knowledge to our new RST parsing task, since the reformulated sequences are quite close to natural language text. Despite the lack of annotated data in the parsing task, our method works well without extra complicated frameworks or features. In the meantime, the subtasks of EDU segmentation and prediction of structure and relations are all integrated into the single process of text generation, which is superior to other approaches in terms of efficiency.

3.4 Postprocessing

In postprocessing, we should first modify and align the output sequence from generation models with the format we design during the linearization. Then an algorithm is executed on the cleaned sequence to restore the node information of the RST parsing trees (the constituent trees).

Clear the format errors Format errors are inevitable since our output sequences directly come from generation models. And the main errors include redundant or lost content, spelling mistakes, and mismatched brackets or relation labels. Considering that the part of the text spans in the output sequence should match the input sentence, we employ an algorithm of Levenshtein Distance based on dynamic programming to modify the output sequence. If the content inside a pair of brackets should be totally deleted, the brackets and following labels will be abandoned together.

Then we calculate the number of brackets, labels and EDUs in the sequence to be processed. EDUs are always inside the innermost brackets. For a binary tree, its reformulated sequence must contain $(2n-1)$ pairs of brackets and $(2n-2)$ pairs of relation labels, if the number of EDUs equals n. When there are more or less close brackets and relation labels, we remove or add the corresponding

Algorithm 1 Restore the constituent tree

Input: Target sequence S, input sentence I

```
Initialization: T = [], nodes = [], i = 0
Seq_unit = S.split(']')
U_k = Seq_unit[k].split('['), 0 ≤ k < len(Seq_unit)
repeat
    if '[' in Seq_unit[i] then
        cur_label = U_{i+1}[0]
        cur_text = U_i[-1]
        push(nodes, (cur_text, cur_label))
    else if len(nodes) > 1 then
        (text_1, label_1) = pop(nodes)
        (text_2, label_2) = pop(nodes)
        push(T, (text_1, label_1, text_2, label_2))
        cur_label = U_{i+1}[0]
        cur_text = text_1 + ' ' + text_2
        push(nodes, (cur_text, cur_label))
    end if
    i = i + 1
until I = top(nodes).text
```

Output: T as the set of connected constituents in the constituent tree

number of them at the end of the sequence. The relation labels added are randomly selected in order not to interfere with the prediction of model. Our algorithm ensures that errors of open brackets will not influence the following restoration.

Restore the constituent tree We implement a recursive algorithm based on the designed format in reformulation to reconstruct the constituent tree through continually merging bottom text spans. More details are shown in Algorithm 1. In our experiments, no more than 4% of the output sequences have format errors, and all of them can be fixed and converted into the sets of connected constituents using our algorithm without ground truth parsing trees.

4 Experiments

In this section, we introduce the dataset and settings in our experiments and present the results of our end-to-end method for both sentence-level RST parsing and discourse segmentation. The improvement of the augmented data we create is demonstrated as well.

4.1 Datasets

We implement our experiments on the RST Discourse TreeBank (Carlson et al., 2001), which is the standard dataset also used by other studies. It is the largest available discourse corpus and contains 385 Wall Street Journal English articles selected from the Penn Treebank (Marcus et al., 1993), 347 for training and 38 for test.

Dataset	#Training	#Test
Doc-level RST-DT	347	38
Sent-level RST-DT	7156	951
Discourse Segmentation	7156	991

Table 1: The statistics of datasets for different tasks.

To construct the dataset for sentence-level RST parsing, we follow the same preprocessing step as Joty et al. (2012); Liu et al. (2019); Lin et al. (2019). We segment sentences from raw text using the nltk tools and then select those that consist of several EDUs and form the subtrees of document-level parsing trees. In all, we obtain 7156 sentences for training, together with their parsing trees, and 906 for test, which is a bit smaller than the scale reported by Lin et al. (2019) (7321 for training and 951 for test). This may be due to the different ways of identifying sentences. Fortunately, we are provided with the test set created by Lin et al. (2019) to replace the one processed by ourselves, ensuring a fair test and comparison.

As for discourse segmentation, we directly use 7156 sentences in the sentence-level RST parsing task for training and the same test set as Lin et al. (2019). Our training set is also smaller compared with the one they used. For both tasks, we randomly select 10% of the training data for hyperparameter tuning. An overview of these datasets is shown in Table 1.

4.2 Model and Settings

In our experiments, we select T5-base (Raffel et al., 2020) as the pretrained model. The family of T5 models is the encoder-decoder model pretrained on various tasks converted into the text-to-text format, which caters to our method. We also attempt the byte-level ByT5 (Xue et al., 2021) and other generative pretrained models, such as BART (Lewis et al., 2020), but they are less effective.

In the training process, we set the batch size to 16, and the maximum input and output sequence length to that of the longest sequence, which is not longer than 512. The training epoch is set to 50 in end-to-end parsing and 40 in experiments with augmented data. The Adamw optimizer is used with a learning rate of 3e-4 together with the cosine learning rate decay scheduler, and the warmup rate is set to 0.1.

During inference, we employ beam search with a beam size of 8 without repetition penalty since our target sequence may contain repeated relation labels and brackets. To achieve stable decoding performance, we average the model parameters over the last five epochs. All the experiments are repeated at least five times with different random seeds, and the average results are reported.

4.3 Evaluation Metric

To evaluate the performance of our method, we follow RST-Parseval metrics (Marcu, 2000), containing micro-averaged F1-scores of unlabeled (Span) and labeled (Nuclearity, Relation). For fair comparison, we use 18 rhetorical relations defined in Carlson and Marcu (2001), same as other sentence-level RST parsing studies (Liu et al., 2019; Lin et al., 2019).

In the task of discourse segmentation, we evaluate the performance only with respect to the intra-sentential segment boundaries and report the results of precision, recall, and micro-averaged F1-score to keep the same with Wang et al. (2018).

4.4 Data Augmentation

Before demonstrating the experiment results, we introduce our data augmentation strategies. The lack of annotated RST parsing trees has been hindering research on discourse parsing since annotators must be experts in discourse analysis and the manual designed for the annotation is quite complicated. From this point, we intend to expand the training set with the augmented data, which is generated and filtered according to our designed rules.

Considering that the RST-DT consists of only a small part of the documents in the WSJ corpus and the rest remain without annotation, we can use them to create silver data which keep the same domain with the RST-DT. First, the documents in the WSJ corpus that are not selected for annotation in RST-DT are extracted and split into sentences similarly. We choose three parsers trained by our end-to-end method with different random seeds and utilize them to generate candidate output sequences for each sentence we have selected. In this way, we can get the initial and promiscuous instances for parsing, each instance with an input sentence and three plausible output sequences.

To obtain the high-quality data, we check these sequences according to the format we design in the reformulation. And the rule of annotation for RST parsing is also taken into consideration. We first discard the sequences that have redundant or missing content compared with their input sentences. Then, if the numbers of EDUs, brackets, and relation labels are not matched, the corresponding sequence will also be abandoned. For the rest of the sequences, we employ Algorithm 1 on each of them to restore the constituent information and check whether the relation labels follow the rule of annotation. When nucleus and satellite relations appear together, they should be assigned the label Span and a rhetorical relation label respectively. And two nucleus relations should use the same relation labels other than the label Span.

Dataset	#Sentence	#Avg EDU	#Avg word
Training set	7156	2.49	21.41
Initial silver data	41833	2.80	26.54
+ content check	39258	2.61	25.37
+ brackets match	37360	2.43	24.29
+ labeling rules	37324	2.42	24.26

Table 2: The statistics of our augmented dataset and original training set.

Through the strategies above, we get those well-formed sequences that follow the labeling rules and have no format errors. If an input sentence still pairs with more than one candidate output sequence, we decide the target sequence via majority voting. The details of our augmented dataset filtered with different strategies are shown in Table 2. It can be found that the average numbers of EDUs and words in the augmented dataset gradually approach those of the training set during filtering, which helps to reduce the distribution difference between the two datasets.

4.5 Experimental Results

We evaluate our method on both tasks: (a) sentence-level RST parsing; (b) discourse segmentation. Benefiting from our end-to-end method, the parsing tree can be directly built from the corresponding input text without using gold EDU segmentation. And the EDU segmentation is predicted simultaneously during parsing and can be extracted from the generated parsing tree as the attached results.

RST parsing Since our end-to-end method unifies the traditional two stages of RST parsing, we compare our results with the models that also do not make use of gold EDU segmentation (Soricut and Marcu, 2003; Joty et al., 2012; Lin et al., 2019).

Approach	S	N	R
Soricut and Marcu (2003)	76.70	70.20	58.00
Joty et al. (2012)	82.40	76.60	67.50
Lin et al. (2019) (Pipeline)	91.14	85.80	76.94
Lin et al. (2019) (Joint)	91.75	86.38	77.52
Our Method			
End-to-end parser	92.88	88.22	80.27
+ data augmentation	**93.27**	**88.70**	**80.89**

Table 3: Results for sentence-level RST parsing without gold EDU segmentation. The columns of S, N and R indicate the micro-averaged F1-scores of Span, Nuclearity and Relation respectively.

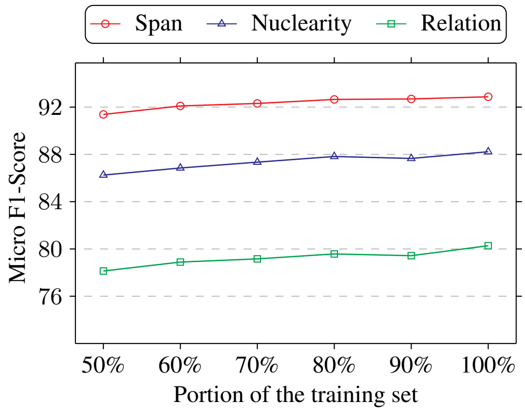


Figure 3: The performance variation curve with different portions of the training set.

Approach	P	R	F1
Human Agreement	98.50	98.20	98.30
Soricut and Marcu (2003)	83.80	86.80	85.20
Fisher and Roark (2007)	91.30	89.70	90.50
Joty et al. (2012)	88.00	92.30	90.10
Li et al. (2018)	91.08	91.03	91.05
Wang et al. (2018)	92.04	94.41	93.21
Lin et al. (2019) (BERT)	92.05	95.03	93.51
Lin et al. (2019) (ELMo)	94.12	96.63	95.35
Lin et al. (2019) (Joint)	93.34	**97.88**	95.55
Our Method			
Extraction from parsing	95.50	96.85	96.17
+ data augmentation	**95.99**	96.64	**96.32**

Table 4: Results for discourse segmentation. The columns of P, R and F1 indicate the Precision, Recall and micro-averaged F1-score respectively.

These methods utilized extra trained automatic segmenters to generate imprecise segmentation and send it to their parsing models to build the parsing tree. Besides the pattern of the pipeline, Lin et al. (2019) proposed jointly training the segmenting and parsing models to further improve the performance on both tasks.

We demonstrate the results in Table 3. The performance of our end-to-end method is substantially better than the existing state-of-the-art model, with the improvement of approximately 1.1, 1.8 and 2.7 absolute points in Span, Nuclearity and Relation respectively. The obvious advancement in Nuclearity and Relation illustrates that the integration of relation labels and input text can be learned more effectively through our reformulation, compared with the traditional form of classification tasks with separate frameworks.

It is also worth noting that the joint model of Lin et al. (2019) utilized the extra instances of the discourse segmentation task, which do not exist in the training set of the RST parsing. Given that our training set is already smaller than theirs, our method achieves better performance with less data. To further explore the influence of the scale of training data, we experiment with 50%, 60%, 70%, 80% and 90% of the training set. The results in Figure 3 show that our method can outperform the state-of-the-art model by only using 60% of the training set. And the performance curve indicates that more instances may still be able to promote the performance of the parser.

Then we combine the original training set with our augmented data and repeat the training process similarly. The results of our end-to-end parser with the help of the augmented data can also be found in Table 3, which get further enhancement of about 0.5 absolute point on all of Span, Nuclearity and Relation.

Discourse segmentation In fact, a parsing tree itself contains the EDU segmentation of the corresponding text because it is EDUs that serve as the leaves of the tree structure. Since we built the parsing tree from the input sentence without gold EDU segmentation, we equivalently perform the segmentation task at the same time through extracting the EDU segmentation from the generated parsing tree. We evaluate the performance and show the results in Table 4.

The performance of segmentation extracted from parsing trees surpasses the best joint model from Lin et al. (2019) in Precision and F1-score. And with the help of augmented data, we get about 2.7 and 0.8 absolute points of increase in Preci-

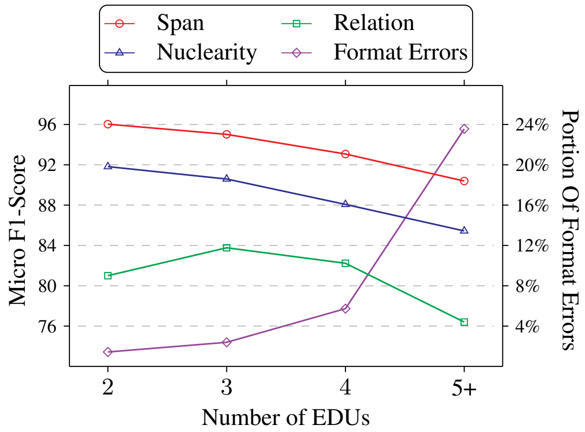


Figure 4: Performances on Span, Nuclearity and Relation, together with the portion of instances containing format errors with different numbers of EDUs.

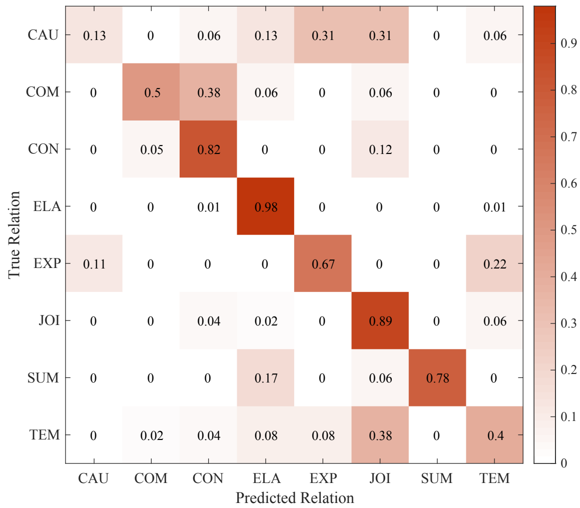


Figure 5: Confusion matrix for eight semantically similar rhetorical relation labels: Cause(CAU), Comparison(COM), Contrast(CON), Elaboration(ELA), Explanation(EXP), Joint(JOI), Summary(SUM), Temporal(TEM).

sion and F1-score, but a 0.5 point drop in Recall compared with the existing state-of-the-art model. With the significant improvement in precision, the segmenter may generate fewer wrong EDUs that do not exist in the gold segmentation set, reducing the error accumulation. Moreover, considering that we use a smaller training set compared with other studies and the existing state-of-the-art model was trained specifically on this task, our method shows superiority in terms of efficiency.

4.6 Error Analysis

In Figure 4, we show the respective performances of instances with different numbers of EDUs. The micro F1-scores of Span and Nuclearity drop as the number of EDUs increases, while Relation achieves a low score when the instance only includes two EDUs. We suppose that the increasing difficulty of parsing longer sentences reduces the performance of our method since it remains a challenging problem for the language model to understand long sequences. In addition, short sentences may not contain sufficient information for the model to infer the Relation label, considering that there are 18 rhetorical relations to be identified, while the nuclearity relations only contain two.

The portion of instances with format errors is also reported in Figure 4. The rapid growth of format errors as the number of EDUs increases shows the difficulty for the model in generating long sequences precisely in keeping with the constraints of our formats. It can also be proven by the decreasing average EDUs of silver data when more filtering rules are added. It is challenging but significant for future research to explore how to improve our end-to-end method when dealing with long sequences since it is the main performance bottleneck.

We also show the confusion matrix for eight semantically similar rhetorical relation labels in Figure 5, some of which are also mentioned in other studies. Our method fails to effectively distinguish between Temporal and Joint, Comparison and Contrast, but succeeds in Explanation and Elaboration. Some examples of our successfully predicted instances and format errors in output sequences can be found in Appendix A and B respectively.

5 Conclusion

In this paper, we propose a simple but effective end-to-end method for sentence-level RST parsing to generate the parsing tree directly from the input text. We convert RST parsing into text-to-text generation by reformulating each parsing tree into an equivalent linear sequence. Benefiting from the latent knowledge in pretrained models, our method does not require additional features or neural frameworks and can simultaneously perform the discourse segmentation during parsing. Experimental results show that our method substantially outperforms existing approaches on both tasks. Furthermore, we create high-quality augmented data to alleviate the lack of annotated RST parsing trees and further improve the performance of our method. In future research, we will explore how to better deal with long sequences and effectively apply our method to document-level RST parsing.

References

Parminder Bhatia, Yangfeng Ji, and Jacob Eisenstein. 2015. Better document-level sentiment analysis from RST discourse parsing. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 2212–2218. The Association for Computational Linguistics.

Lynn Carlson and Daniel Marcu. 2001. Discourse tagging reference manual. *ISI Technical Report ISI-TR-545*, 54(2001):56.

Lynn Carlson, Daniel Marcu, and Mary Ellen Okurovsky. 2001. Building a discourse-tagged corpus in the framework of rhetorical structure theory. In *Proceedings of the SIGDIAL 2001 Workshop, The 2nd Annual Meeting of the Special Interest Group on Discourse and Dialogue, Saturday, September 1, 2001 to Sunday, September 2, 2001, Aalborg, Denmark*. The Association for Computer Linguistics.

Vanessa Wei Feng and Graeme Hirst. 2014. A linear-time bottom-up discourse parser with constraints and post-editing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014, June 22-27, 2014, Baltimore, MD, USA, Volume 1: Long Papers*, pages 511–521. The Association for Computer Linguistics.

Seeger Fisher and Brian Roark. 2007. The utility of parse-derived features for automatic discourse segmentation. In *ACL 2007, Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics, June 23-30, 2007, Prague, Czech Republic*. The Association for Computational Linguistics.

Shima Gerani, Yashar Mehdad, Giuseppe Carenini, Raymond T. Ng, and Bita Nejat. 2014. Abstractive summarization of product reviews using discourse structure. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1602–1613. ACL.

Hugo Hernault, Helmut Prendinger, David A. duVerle, and Mitsuru Ishizuka. 2010. HILDA: A discourse parser using support vector machine classification. *Dialogue Discourse*, 1(3):1–33.

Peter Jansen, Mihai Surdeanu, and Peter Clark. 2014. Discourse complements lexical semantics for non-factoid answer reranking. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014, June 22-27, 2014, Baltimore, MD, USA, Volume 1: Long Papers*, pages 977–986. The Association for Computer Linguistics.

Yangfeng Ji and Jacob Eisenstein. 2014. Representation learning for text-level discourse parsing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014, June 22-27, 2014, Baltimore, MD, USA, Volume 1: Long Papers*, pages 13–24. The Association for Computer Linguistics.

Shafiq R. Joty, Giuseppe Carenini, and Raymond T. Ng. 2012. A novel discriminative framework for sentence-level discourse analysis. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, EMNLP-CoNLL 2012, July 12-14, 2012, Jeju Island, Korea*, pages 904–915. ACL.

Shafiq R. Joty, Giuseppe Carenini, Raymond T. Ng, and Yashar Mehdad. 2013. Combining intra- and multi-sentential rhetorical parsing for document-level discourse analysis. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, ACL 2013, 4-9 August 2013, Sofia, Bulgaria, Volume 1: Long Papers*, pages 486–496. The Association for Computer Linguistics.

Shafiq R. Joty, Francisco Guzmán, Lluís Màrquez, and Preslav Nakov. 2017. Discourse structure in machine translation evaluation. *Comput. Linguistics*, 43(4).

Naoki Kobayashi, Tsutomu Hirao, Hidetaka Kamigaito, Manabu Okumura, and Masaaki Nagata. 2020. Top-down RST parsing utilizing granularity levels in documents. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 8099–8106. AAAI Press.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 7871–7880. Association for Computational Linguistics.

Jing Li, Aixin Sun, and Shafiq R. Joty. 2018. Segbot: A generic neural text segmentation model with pointer network. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*, pages 4166–4172. ijcai.org.

Jiwei Li, Rumeng Li, and Eduard H. Hovy. 2014. Recursive deep models for discourse parsing. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 2061–2069. ACL.

Xiang Lin, Shafiq R. Joty, Prathyusha Jwalapuram, and Saiful Bari. 2019. A unified linear-time framework for sentence-level discourse parsing. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July*

28- August 2, 2019, Volume 1: Long Papers, pages 4190–4200. Association for Computational Linguistics.

Linlin Liu, Xiang Lin, Shafiq R. Joty, Simeng Han, and Lidong Bing. 2019. Hierarchical pointer net parsing. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 1007–1017. Association for Computational Linguistics.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *CoRR*, abs/2107.13586.

Annie Louis, Aravind K. Joshi, and Ani Nenkova. 2010. Discourse indicators for content selection in summarization. In *Proceedings of the SIGDIAL 2010 Conference, The 11th Annual Meeting of the Special Interest Group on Discourse and Dialogue, 24-15 September 2010, Tokyo, Japan*, pages 147–156. The Association for Computer Linguistics.

William C Mann and Sandra A Thompson. 1987. *Rhetorical structure theory: A theory of text organization*.

Daniel Marcu. 2000. The rhetorical parsing of unrestricted texts: A surface-based approach. *Comput. Linguistics*, 26(3):395–448.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of english: The penn treebank. *Comput. Linguistics*, 19(2):313–330.

Thanh-Tung Nguyen, Xuan-Phi Nguyen, Shafiq R. Joty, and Xiaoli Li. 2021. RST parsing from scratch. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 1613–1625. Association for Computational Linguistics.

Giovanni Paolini, Ben Athiwaratkun, Jason Krone, Jie Ma, Alessandro Achille, Rishita Anubhai, Cícero Nogueira dos Santos, Bing Xiang, and Stefano Soatto. 2021. Structured prediction as translation between augmented natural languages. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Livia Polanyi and Martin van den Berg. 2011. Discourse structure and sentiment. In *Data Mining Workshops (ICDMW), 2011 IEEE 11th International Conference on, Vancouver, BC, Canada, December 11, 2011*, pages 97–102. IEEE Computer Society.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67.

Radu Soricut and Daniel Marcu. 2003. Sentence level discourse parsing using syntactic and lexical information. In *Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics, HLT-NAACL 2003, Edmonton, Canada, May 27 - June 1, 2003*. The Association for Computational Linguistics.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 2692–2700.

Yizhong Wang, Sujian Li, and Houfeng Wang. 2017. A two-stage parsing method for text-level discourse analysis. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 2: Short Papers*, pages 184–188. Association for Computational Linguistics.

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, Brussels, Belgium, October 31 - November 4, 2018*, pages 962–967. Association for Computational Linguistics.

Linting Xue, Aditya Barua, Noah Constant, Rami Al-Rfou, Sharan Narang, Mihir Kale, Adam Roberts, and Colin Raffel. 2021. Byt5: Towards a token-free future with pre-trained byte-to-byte models. *CoRR*, abs/2105.13626.

Nan Yu, Meishan Zhang, and Guohong Fu. 2018. Transition-based neural RST parsing with implicit syntax features. In *Proceedings of the 27th International Conference on Computational Linguistics, COLING 2018, Santa Fe, New Mexico, USA, August 20-26, 2018*, pages 559–570. Association for Computational Linguistics.

Longyin Zhang, Yuqing Xing, Fang Kong, Peifeng Li, and Guodong Zhou. 2020. A top-down neural architecture towards text-level parsing of discourse rhetorical structure. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 6386–6395. Association for Computational Linguistics.

Ying Zhang, Hidetaka Kamigaito, and Manabu Okumura. 2021. A language model-based generative classifier for sentence-level discourse parsing. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP*

2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021, pages 2432–2446. Association for Computational Linguistics.

A Example Demonstration

Figure 6 shows an instance mistakenly labeled Summary as Elaboration by the other parser Nguyen et al. (2021), but is successfully predicted by our method. We also demonstrate the corresponding output sequence from our method together with the restored parsing tree and the extracted EDU segmentation.

B Format Errors

Figure 7 shows some example format errors from our generated output sequences.

(a) Input Sentence

The natural resources development concern said proceeds will be used to repay long-term debt, which stood at 598 million Canadian dollars (US$510.6 million) at the end of 1988.

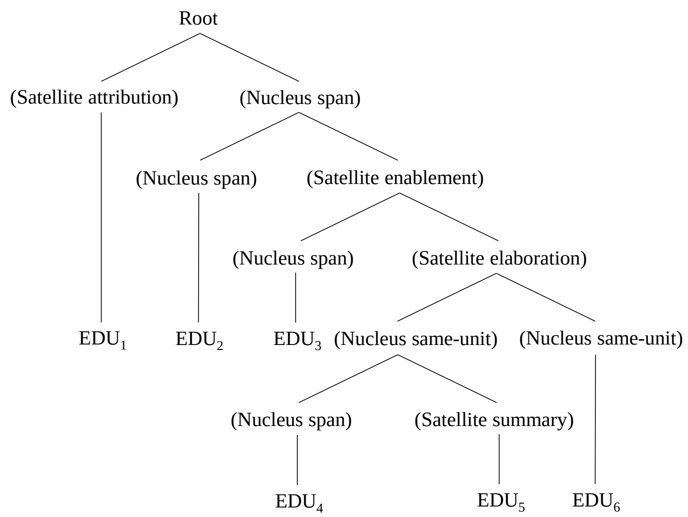
(b) Output Sequence

[[The natural resources development concern said] Satellite attribution [[proceeds will be used] Nucleus span [[to repay long-term debt,] Nucleus span [[[which stood at 598 million Canadian dollars] Nucleus span [(US$510.6 million)] Satellite summary] Nucleus same-unit [at the end of 1988.] Nucleus same-unit] Satellite elaboration] Satellite enablement] Nucleus span]

(c) Restored Constituents

(which stood at 598 million Canadian dollars Nucleus span (US$510.6 million) Satellite summary)
(which stood at 598 million Canadian dollars (US$510.6 million) Nucleus same-unit at the end of 1988. Nucleus same-unit)
(to repay long-term debt, Nucleus span which stood at 598 million Canadian dollars (US$510.6 million) at the end of 1988. Satellite elaboration)
(proceeds will be used Nucleus span to repay long-term debt, which stood at 598 million Canadian dollars (US$510.6 million) at the end of 1988. Satellite enablement)
(The natural resources development concern said Satellite attribution proceeds will be used to repay long-term debt, which stood at 598 million Canadian dollars (US$510.6 million) at the end of 1988. Nucleus span)

(d) Parsing Tree



(e) EDU Segmentation

EDU$_1$: The natural resources development concern said
EDU$_2$: proceeds will be used
EDU$_3$: to repay long-term debt,
EDU$_4$: which stood at 598 million Canadian dollars
EDU$_5$: (US$510.6 million)
EDU$_6$: at the end of 1988.

(f) Mistaken Label

Figure 6: An example of the output sequence and postprocessing using our method. The red part shows we correctly predict Summary while the other parser mistakenly labels Elaboration. The blue part represents the labels for the text spans before them.

(a) Matching Errors

[[["Oh, I bet] Satellite attribution [it'll be up 50 points on Monday,"] Nucleus span] Nucleus span [said Lucy Crump, a 78-year-old retired housewife in Lexington, Ky.] Satellite attribution]

[[An interest rate is guaranteed for between one and seven years,] Nucleus span [[after which holders get 30 days] Nucleus span [[to choose another guarantee period or to switch to another insurer's contract] Nucleus span [[without the surrender charges] Nucleus span [that are common to annuities.] Satellite elaboration] Satellite elaboration] Satellite temporal] Satellite contrast]

(b) Content Errors

[[Everytime (Every time) he sees me,] Satellite background [he gets very nervous."] Nucleus span]

Figure 7: Several examples of format errors in output sequences. The red part is missed and the blue part is the true content in the corresponding input sentence.