Leveraging Plug-and-Play Models for Rhetorical Structure Control in Text Generation

Yuka Yokogawa[1], Tatsuya Ishigaki[2], Hiroya Takamura[2],
Yusuke Miyao[2,3], Ichiro Kobayashi[1,2]
g1820542@is.ocha.ac.jp, {ishigaki.tatsuya, takamura.hiroya}@aist.go.jp,
yusuke@is.s.u-tokyo.ac.jp, koba@is.ocha.ac.jp
[1]Ochanomizu University, Japan,
[2]National Institute of Advanced Industrial Science and Technology, Japan,
[3]University of Tokyo, Japan

Abstract

We propose a method that extends a BART-based language generator using the plug-and-play language model to control the rhetorical structure of generated text. Our approach considers rhetorical relations between clauses and generates sentences that reflect this structure using plug-and-play language models. We evaluated our method using the Newsela corpus, which consists of texts at various levels of English proficiency. Our experiments demonstrated that our method outperforms the vanilla BART in terms of the correctness of output discourse and rhetorical structures. In existing methods, the rhetorical structure tends to deteriorate when compared to the baseline, the vanilla BART, as measured by n-gram overlap metrics such as BLEU. However, our proposed method does not exhibit this significant deterioration, demonstrating its advantage.

1 Introduction

Language generation technology has been significantly improved due to the advance of pre-trained language models. However, although we would often like to have a text with a certain discourse or logical structure, the current technology has difficulty in following such global constraints. In this paper, we address the task of controlling natural language generation in terms of the discourse structure of the generated text.

As a discourse structure, we employ a tree structure based on Rhetorical Structure Theory (RST) (Mann and Thompson, 1988). An existing work RSTGen (Adewoyin et al., 2022) incorporates rhetorical structures into text generation by transforming trees into embeddings prior to generation. In contrast to RSTGen, our method dynamically controls the rhetorical structure during text generation using the plug-and-play language model (PPLM) (Dathathri et al., 2019), as shown in Figure 1. One significant advantage of our method is that fine-tuning of the language model for RST is not necessary. PPLM was originally designed to control the topic of the generated text with the help of a topic classifier. In our method, rhetorical relations are regarded as topics, and a classifier identifying the rhetorical relationship between text segments is employed instead of a topic classifier.

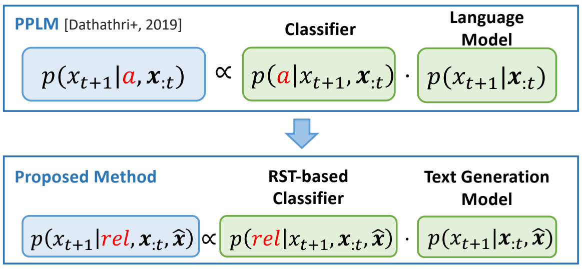
PPLM [Dathathri+, 2019]
Classifier
Language Model
$p(x_{t+1}|a, \boldsymbol{x}_{:t}) \propto p(a|x_{t+1}, \boldsymbol{x}_{:t}) \cdot p(x_{t+1}|\boldsymbol{x}_{:t})$
Proposed Method
RST-based Classifier
Text Generation Model
$p(x_{t+1}|rel, \boldsymbol{x}_{:t}, \hat{\boldsymbol{x}}) \propto p(rel|x_{t+1}, \boldsymbol{x}_{:t}, \hat{\boldsymbol{x}}) \cdot p(x_{t+1}|\boldsymbol{x}_{:t}, \hat{\boldsymbol{x}})$

Figure 1: Formulating a task for RST-based text generation. Our model is based on the plug-and-play language model (PPLM) that controls language models to generate texts with a specific attribute a. We consider the relation label in RST (Mann and Thompson, 1988) as the desired attribute.

We evaluate our method on the Newsela corpus (Xu et al., 2015), a dataset, which consists of texts at various levels of English proficiency. Our experiments demonstrated that our method outperforms the vanilla BART baseline in terms of the correctness of output rhetorical structures. In existing methods, the rhetorical structure tends to deteriorate when compared to the baseline, as measured by n-gram overlap metrics such as BLEU (Papineni et al., 2002). However, our proposed method does not exhibit this significant deterioration, demonstrating an advantage.

2 Related Work

Rhetorical Structure Theory (RST) (Mann and Thompson, 1988) represents the semantic relationships within a text as a constituency binary tree, while a dependency tree-based framework (Prasad

Proceedings of the 17th International Natural Language Generation Conference, pages 486–493
September 23–27, 2024. ©2024 Association for Computational Linguistics

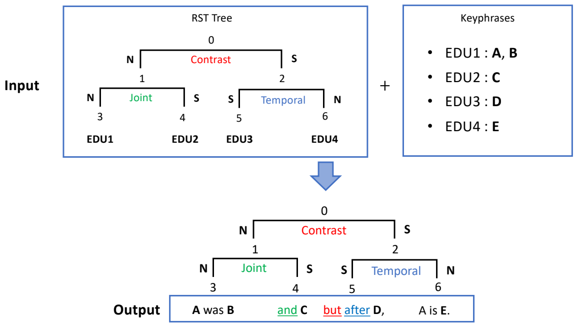

Figure 2: An example of input and output. The input consists of a binary RST tree and keyphrases (important words or phrases). The output is a token sequence reflecting the specified RST tree and keyphrases.

et al., 2008) also exists. A recent study proposed to incorporate discourse structures into a language model using Variational Auto Encoder (Ji and Huang, 2021). We use RST by following recent works in generation (Adewoyin et al., 2022; Ji and Huang, 2021). Early approaches treated the incorporation of RST into text generation as a planning problem (Hovy, 1988; Hovy and McCoy, 2014). Integrating tree structure into neural network-based language generators has been actively studied. Adewoyin et al. (2022) incorporated RST trees into an autoregressive language model by converting them into embeddings. Chernyavskiy (2022) created the entire text plan as an RST tree, followed by autoregressive generation of the text span using a language model. In contrast to the aforementioned works, our method dynamically controls rhetorical structure during text generation using PPLM.

3 RST-based Controllable Generation

For our experiments, we utilize a binary form of RST tree following RSTGen (Adewoyin et al., 2022). To construct a binary form of the RST tree from a text, the text is divided into smaller units, called Discourse Units (DUs). We assign an index to each node in the tree, starting from zero. When the index of a parent node is i, the left child node is indexed as $2i+1$, and the right child node as $2i+2$. The text at a node with no children represents Elementary Discourse Unit (EDU), and the text at a parent node corresponds to a pair of DUs. A parent node has a relationship label and a nuclearity label indicating the semantic relationship of sibling DUs.

Task Formulations We formulate the controlling text generation based on RST as a conditional text generation. The input consists of a binary RST tree, keyphrases, and their positions in the tree. In this paper, a binary RST tree is represented by a sequence of relation labels $\boldsymbol{rel} = (rel_0, \ldots, rel_N)$. For instance, the RST tree in Figure 2 is encoded as $\boldsymbol{rel} = (\text{Joint}, \text{Contrast}, \text{Temporal})$. Keyphrases are represented as $\hat{\boldsymbol{x}}$. It is a reference token sequence all replaced by masks except the positions of keyphrases and the special token that indicates the EDU delimiter. The position of the keyphrases, although typically a training target, is assumed known in this study to focus only on RST-based control. The output is a token sequence $\boldsymbol{x}$ reflecting the inputs. In this paper, we formulate the generation of a token within an EDU conditioned on the specified relation label as follows:

$$x_{t+1} \sim p(x_{t+1}|rel, \boldsymbol{x}_{:t}, \hat{\boldsymbol{x}}) \tag{1}$$

3.1 Control with Classifier

Our approach to controlling text generation relies on the plug-and-play language model(PPLM) (Dathathri et al., 2019). Let a denote an attribute to be introduced. The goal of controllable text generation is to model the distribution $p(\boldsymbol{x}|a)$. PPLM models this distribution by multiplying $p(a|\boldsymbol{x})$ with $p(\boldsymbol{x})$ according to Bayes' theorem: $p(\boldsymbol{x}|a) \propto p(a|\boldsymbol{x}) \cdot p(\boldsymbol{x})$. A classifier defines the distribution $p(a|\boldsymbol{x})$.

Building on the concept of PPLM, we propose a method to control the text generator to generate a text reflecting specified relation labels using the classifier that identifies the relation labels between EDUs. We model the desired distribution (on the right-hand side of Equation (1) by multiplying the distribution represented by the generator with the distribution represented by the classifier:

$$\begin{aligned} & p(x_{t+1}|rel, \boldsymbol{x}_{:t}, \hat{\boldsymbol{x}}) \\ & \quad \propto \quad p(rel|x_{t+1}, \boldsymbol{x}_{:t}, \hat{\boldsymbol{x}}) \cdot p(x_{t+1}|\boldsymbol{x}_{:t}, \hat{\boldsymbol{x}}) \end{aligned} \tag{2}$$

Generator We train an encoder-decoder language model to generate text based on provided keyphrases for each EDU. The token sequence representing keyphrases and their positions $\hat{\boldsymbol{x}}$ is encoded, and the decoder generates the token sequence $\boldsymbol{x}$ autoregressively. This model can be denoted as a language model that represents the following distribution: $p(x_{t+1}|\boldsymbol{x}_{:t}, \hat{\boldsymbol{x}})$.

Classifier We introduce a classifier that identifies the relation label between a pair of EDUs. The input is a pair of EDUs, and the output is a relation

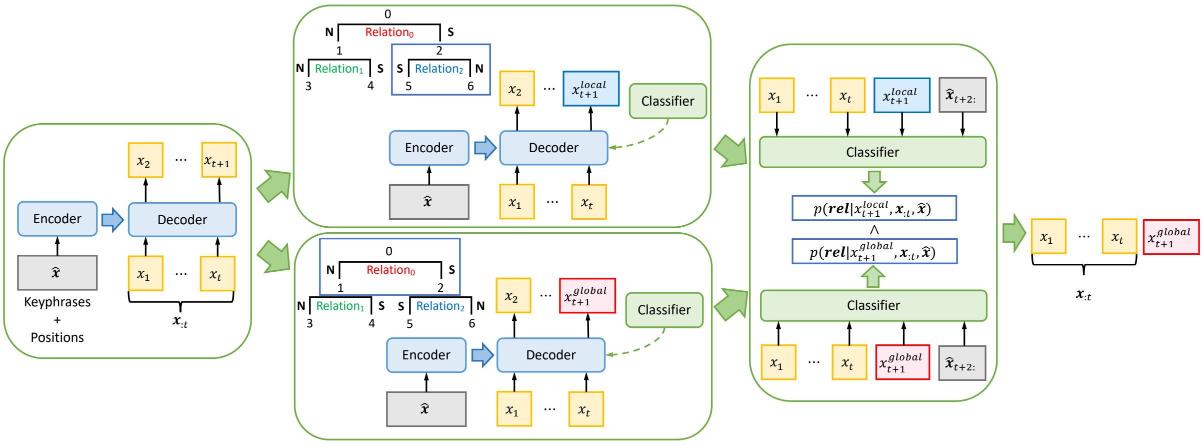
Encoder
Decoder
$\hat{\boldsymbol{x}}$
Keyphrases
+
Positions
x_2 ··· x_{t+1}
x_1 ··· x_t
$\boldsymbol{x}_{:t}$
0
N Relation₀ S
1
2
N Relation₁ S
S Relation₂ N
3 4 5 6
x_2 ··· x_{t+1}^{local}
x_2 ··· x_{t+1}^{global}
Classifier
Classifier
x_1 ··· x_t x_{t+1}^{local} $\hat{\boldsymbol{x}}_{t+2:}$
$p(\boldsymbol{rel}|x_{t+1}^{local}, \boldsymbol{x}_{:t}, \hat{\boldsymbol{x}})$
$\wedge$
$p(\boldsymbol{rel}|x_{t+1}^{global}, \boldsymbol{x}_{:t}, \hat{\boldsymbol{x}})$
x_1 ··· x_t x_{t+1}^{global} $\hat{\boldsymbol{x}}_{t+2:}$
x_1 ··· x_t x_{t+1}^{global}
$\boldsymbol{x}_{:t}$

Figure 3: The process of obtaining an output token at each time-step. At first, the generator produces a token without any control. Subsequently, the generation process is controlled by the classifier to reflect the relation label. Given the hierarchical nature of rhetorical structures, tokens are generated with a control based on the relationships at the lower and higher levels. These tokens serve as candidates for the final output, determined by comparing their probabilities calculated by the classifier.

label rel. This classifier represents the following distribution: $p(rel|x_{t+1}, \boldsymbol{x}_{:t}, \hat{\boldsymbol{x}})$.

3.2 Hierarchy-aware Generation

The token x_{t+1} at time-step $t+1$ is obtained through the following procedure3: (1) The generator produces an output token without any control (at the left of figure 3). (2) Considering the hierarchical nature of rhetorical structures, we consider two relationships for the EDU containing the output token, the relationship at the lower and higher levels. At each level, we generate a token with additional control based on the relation label (at the center of figure 3). These two outputs become candidates for the final output. (3) We calculate the probability of each token sequence, including the respective candidates, having the specified sequence of relation labels using the classifier. We choose the one with the higher probability as the final output (at the right of figure 3).

For example, we generate the token x_{t+1} in the third EDU (EDU3) in Figure 2 from the state where EDU1 and EDU2 have been generated. First, the generator outputs a token x_{t+1} without any control: $x_{t+1} \sim p(x_{t+1}|\boldsymbol{x}_{:t}, \hat{\boldsymbol{x}})$. Next, we control the generation by the generator to reflect relation labels using the classifier. From the hierarchy of rhetorical structures, we can consider two relationships for the EDU containing the output token, the relationship at the lower level of the hierarchy and the relationship at the higher level. For EDU3 in Figure 2, the relationship at the lower level is "Temporal" with EDU4, and we call it as the local relationship. In the same way, the relationship at the higher level is "Contrast" with the pair of EDU1 and EDU2, and we call it as the global relationship. For each of these two levels, an output token is obtained based on the respective relationships. Let the relation label rel in Equation (2) be "Temporal" and the input of the classifier be the pairs of EDU3 and EDU4, one output token x_{t+1}^{local} is obtained : $x_{t+1}^{local} \sim p(x_{t+1}|\text{Temporal}, \boldsymbol{x}_{:t}, \hat{\boldsymbol{x}})$. In the same way, the other candidate x_{t+1}^{global} is obtained based on the "Contrast" relationship : $x_{t+1}^{global} \sim p(x_{t+1}|\text{Contrast}, \boldsymbol{x}_{:t}, \hat{\boldsymbol{x}})$. For the token sequence $\boldsymbol{x}_{:t}$ generated up to time-step t, we consider adding each of the two candidate tokens to it. We insert the two candidate tokens, x_{t+1}^{local} and x_{t+1}^{global}, obtained in the previous step into the token sequence $\boldsymbol{x}_{:t}$. Next, we calculate the probability distribution of the sequence of relation labels for the added token sequence by applying the classifier to pairs of EDUs. The input is a token sequence, and the output is a sequence of relation labels $\boldsymbol{rel}$: $p(\boldsymbol{rel}|x_{t+1}, \boldsymbol{x}_{:t}, \hat{\boldsymbol{x}})$ We choose the candidate with the higher probability as the final output x_{t+1}.

4 Experimental Setup

The Newsela corpus (Xu et al., 2015) consists of news articles for readers with various English proficiency levels. Paragraphs extracted from these articles are utilized as the dataset in this paper. We employ a trained RST parser (Kobayashi et al., 2022) to parse each of the dataset. We extract keyphrases using the trained TopicRank keyphrase

Model	Control Method	B-4↑	R-L↑	MTR↑	B-S↑	PPL↓	DM↑	Grammar↑	Redundancy↑	Focus↑	Coherence↑
BART	Keyphrase Positions	60.34	73.93	75.16	95.41	175.60	50.11	64.62	-0.20	-0.00	-21.23
	+ DST-VAE	40.41	61.32	63.31	93.03	148.06	24.20	63.09	-0.33	-0.00	-18.43
	+ RST-Embedding	50.32	69.12	69.95	94.58	190.77	44.84	64.67	-0.02	-0.00	-20.80
	+ RST-PPLM (Ours)	60.16	73.73	74.82	95.37	169.57	50.90	64.33	-0.24	-0.00	-21.95

Table 1: Experimental results on the dataset extracted from Newsela corpus (Xu et al., 2015). DST-VAE is based on DiscoDVT (Ji and Huang, 2021) and RST-Embedding is based on RSTGen (Adewoyin et al., 2022).

extractor (Bougouin et al., 2013). The dataset consists of 25,173, 3,108, and 3,131 samples for training, validation, and testing, respectively.

We used PyTorch library (Paszke et al., 2019) for the implementation. The baseline model was trained by fine-tuning BART (Lewis et al., 2020). AdamW (Loshchilov and Hutter, 2019) was used as the optimization method, and the parameters are included in the appendix. We introduced early stopping when the validation loss did not decrease for three epochs.

We use BART, trained to generate text conditioned on the information of keyphrases and their positions, as the baseline model. We compare our model with two models; 1) DiscoDVT (Ji and Huang, 2021) is a discourse structure-based text generation model. DiscoDVT uses a discrete Variational Auto Encoder, reflecting discourse structures into BART (Lewis et al., 2020). 2) RSTGen (Adewoyin et al., 2022) introduces additional embedding layers for representing RST trees. Embeddings of an RST tree are added to token embeddings, which serve as inputs to language models. We use the RST embeddings from RSTGen as prefix embeddings for the baseline model.

To assess whether the generated texts have specified rhetorical structures, we use the Standard Parse-Eval (Morey et al., 2017) metric. This metric measures how well a labeled tree matches the reference tree in terms of span units. First, we parsed the generated texts using the same RST parser used for annotating the dataset to obtain RST trees. Next, we converted the RST trees into a right-heavy binary structure following (Sagae and Lavie, 2005). Span, Nuclearity, Relation, and Full refer to evaluations of unlabeled, nuclearity-labeled, relation-labeled, and fully labeled tree structures, respectively. We also use BLEU (Papineni et al., 2002), ROUGE (Lin and Hovy, 2003), and METEOR (MTR) (Banerjee and Lavie, 2005) as evaluation metrics. These metrics evaluate the quality of the generated texts by comparing n-gram overlaps with reference texts. BLEU measures precision of n-gram, whereas ROUGE measures recall. METEOR considers both precision and recall. We report BLEU-4 (B-4), which evaluates the overlap of 4-grams, and ROUGE-L (R-L), which measures the longest common subsequence between the generated texts and reference texts. BERTscore (B-S) (Zhang et al., 2020) is used for evaluating semantic similarities. Fluency is evaluated through perplexity (PPL) computed using the medium model of GPT-2 (Radford et al., 2019). Coherence of generated texts is evaluated using two sets of metrics. Firstly, we measure the recall of discourse markers (DM). Discourse markers are words which semantically connect sentences. The recall represents the percentage of correctly generated markers present in the references. Additionally, GRUEN (Zhu and Bhat, 2020) is used. This metric assesses generated texts from following for perspectives: grammaticality, non-redundancy, focus, and coherence.

Model	Span	Nuclearity	Relation	Full
BART	79.08	65.94	56.69	56.33
+DST-VAE	71.03	50.03	36.78	36.41
+RST-Emb	76.29	60.74	50.52	50.03
+RST-PPLM	**82.61**	**69.57**	**60.47**	**60.06**

Table 2: Results based on Standard-Parseval.

5 Results

Table 2 demonstrates that our model (+RST-PPLM) achieves higher scores on Standard-Parseval, which suggests that more texts with correct rhetorical structures are produced.

Table 1 demonstrates that our model achieves closer scores to the baseline in terms of all metrics while other compared models (+DST-VAE and +RST-Embedding) obtained lower scores. For example, DST-VAE achieves only 40.41 in terms of BLEU while our model (+RST-PPLM) and the BART baseline achieve 60.16 and 60.34, respectively. The results suggest that our proposed method does not exhibit this significant deterioration in terms of reference-based metrics.

6 Limitations

In our experiments, we used RST trees with depths of two or less. Thus, our method primarily considers shallow relationships. In contrast, RSTGen imposes a limit of twelve or less levels of tree depth, allowing our proposed method to handle a smaller range of depths. We aim to explore the application of our method to deeper trees.

7 Conclusion

We proposed a method for controllable text generation by language models based on rhetorical structures, inspired by PPLM. While our model did not improve accuracy compared to the baseline, it showed improvement over prior models based on discourse and rhetorical structures. Additionally, we evaluated text coherence in terms of discourse markers and generally observed improved accuracy. However, the depth of the RST tree considered in this paper is limited. Thus, we will extend the proposed model to deeper trees.

8 Applicability to LLMs

This study employed BART as the baseline language model. Proposed method can be applied to recent LLMs under certain conditions. As detailed in the AppendixC, access to both hidden states and logit vectors is necessary for controlling the output using PPLM. Therefore, proposed model also requires access to the model's hidden states and logit vectors. As an example, LLaMA (Touvron et al., 2023) provides access to these components, so our method is likely applicable to it. Future work will involve evaluating the accuracy of the proposed method when applied to LLaMA.

References

Rilwan Adewoyin, Ritabrata Dutta, and Yulan He. 2022. RSTGen: Imbuing fine-grained interpretable control into long-FormText generators. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1822–1835, Seattle, United States. Association for Computational Linguistics.

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, Michigan. Association for Computational Linguistics.

Adrien Bougouin, Florian Boudin, and Béatrice Daille. 2013. TopicRank: Graph-based topic ranking for keyphrase extraction. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 543–551, Nagoya, Japan. Asian Federation of Natural Language Processing.

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 Second SIGdial Workshop on Discourse and Dialogue*.

Alexander Chernyavskiy. 2022. Improving text generation via neural discourse planning. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*, WSDM '22, page 1543–1544, New York, NY, USA. Association for Computing Machinery.

Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. 2019. Plug and play language models: A simple approach to controlled text generation. In *International Conference on Learning Representations*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Eduard H. Hovy. 1988. Planning coherent multisentential text. In *26th Annual Meeting of the Association for Computational Linguistics*, pages 163–169, Buffalo, New York, USA. Association for Computational Linguistics.

Eduard H Hovy and Kathleen F McCoy. 2014. Focusing your rst: A step toward generating coherent multisentential text. In *11th Annual Conference Cognitive Science Society Pod*, pages 667–674. Psychology Press.

Haozhe Ji and Minlie Huang. 2021. DiscoDVT: Generating long text with discourse-aware discrete variational transformer. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4208–4224, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Naoki Kobayashi, Tsutomu Hirao, Hidetaka Kamigaito, Manabu Okumura, and Masaaki Nagata. 2022. A simple and strong baseline for end-to-end neural RST-style discourse parsing. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 6725–6737, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

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*, pages 7871–7880, Online. Association for Computational Linguistics.

Chin-Yew Lin and Eduard Hovy. 2003. Automatic evaluation of summaries using n-gram co-occurrence statistics. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 150–157.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

William C. Mann and Sandra A. Thompson. 1988. Rhetorical structure theory: Toward a functional theory of text organization. *Text & Talk*, 8:243 – 281.

Mathieu Morey, Philippe Muller, and Nicholas Asher. 2017. How much progress have we made on RST discourse parsing? a replication study of recent results on the RST-DT. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1319–1324, Copenhagen, Denmark. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Rashmi Prasad, Nikhil Dinesh, Alan Lee, Eleni Miltsakaki, Livio Robaldo, Aravind Joshi, and Bonnie Webber. 2008. The Penn Discourse TreeBank 2.0. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco. European Language Resources Association (ELRA).

Rashmi Prasad, Eleni Miltsakaki, Nikhil Dinesh, Alan Lee, Aravind K. Joshi, Livio Robaldo, and Bonnie L. Webber. 2007. The penn discourse treebank 2.0 annotation manual.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Kenji Sagae and Alon Lavie. 2005. A classifier-based parser with linear run-time complexity. In *Proceedings of the Ninth International Workshop on Parsing Technology*, pages 125–132, Vancouver, British Columbia. Association for Computational Linguistics.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. Llama: Open and efficient foundation language models. *Preprint*, arXiv:2302.13971.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi 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 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.

Wei Xu, Chris Callison-Burch, and Courtney Napoles. 2015. Problems in current text simplification research: New data can help. *Transactions of the Association for Computational Linguistics*, 3:283–297.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. *Preprint*, arXiv:1904.09675.

Wanzheng Zhu and Suma Bhat. 2020. GRUEN for evaluating linguistic quality of generated text. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 94–108, Online. Association for Computational Linguistics.

A Parameters

We use the learning rate lr $= 5 \times 10^{-5}$, weight_decay $= 0.0$, smoothing value $\epsilon = 1 \times 10^{-8}$. The maximum number of training epochs was set to 20.

B Classifier Experiments

Input and Output The input consists of a pair of EDUs, one being Nucleus and the other Satellite, with the output being a relation label.

Dataset The RST-DT dataset (Carlson et al., 2001) comprises annotated news articles from which EDU pairs, including Nucleus and Satellite, are extracted for our dataset.

Experimental Setups Table 3 shows the experimental setup. We use BART (Lewis et al., 2020) as the language model, and for comparison, we also conduct experiments in the same setting with BERT (Devlin et al., 2019).

Pre-trained model	facebook/bart-base
Training epochs	20
Optimizer	AdamW
Batch size	Train:10,Valid:5,Test:4
Loss function	cross entropy loss
Learning rate	5×10^{-5}

Table 3: Experimental setups.

Model	Accuracy	F1
BERT	55.17	37.59
BART	54.53	37.70

Table 4: Experimental results.

Results Table 4 shows that the BART-based classifier outperforms BERT in the F1 score, although it is inferior to BERT in the accuracy.

C Implementation Details of PPLM

In an efficient implementation of the Transformer (Wolf et al., 2020), the language model's internal states H_t are utilized as inputs when outputting the token x_{t+1} at time-step $t+1$ conditioned on the output token sequence $\boldsymbol{x}_{:t}$ up to time-step t.

$$o_{t+1}, H_{t+1} = \text{LM}(x_t, H_t) \quad (3)$$

$$x_{t+1} \sim p_{t+1} = \text{Softmax}(W o_{t+1}) \quad (4)$$

Here, the internal states is a matrix that retains Key-Value information used in the attention calculation of the Transformer model. PPLM utilizes the gradient from an attribute model $p(a|X)$ to update the internal states, reflecting attribute a.

$$\Delta H_t \leftarrow \Delta H_t + \alpha \frac{\nabla_{\Delta H_t} \log p(a|H_t + \Delta H_t)}{||\nabla_{\Delta H_t} \log p(a|H_t + \Delta H_t)||^{\gamma}} \quad (5)$$

Using the updated internal states $\tilde{H}t = H_t + \Delta H_t$, the language model generates $\tilde{x}t + 1$ based on the token sequence $\boldsymbol{x}$: t up to time-step t.

$$\tilde{o}_{t+1}, H_{t+1} = \text{LM}(x_t, \tilde{H}_t) \quad (6)$$

$$\tilde{x}_{t+1} \sim \tilde{p}_{t+1} = \text{Softmax}(W \tilde{o}_{t+1}) \quad (7)$$

D Results on Recalls

Figure 4 demonstrates that our model significantly improved accuracy for discourse markers like 'since' and 'before', while showing only a slight improvement for 'and' and 'for'. While the former words are closely tied to specific relation labels, the latter are commonly used in text and have weaker associations with relation labels. Consequently, the control based on relation labels proposed in this paper yields a smaller improvement for the latter words.

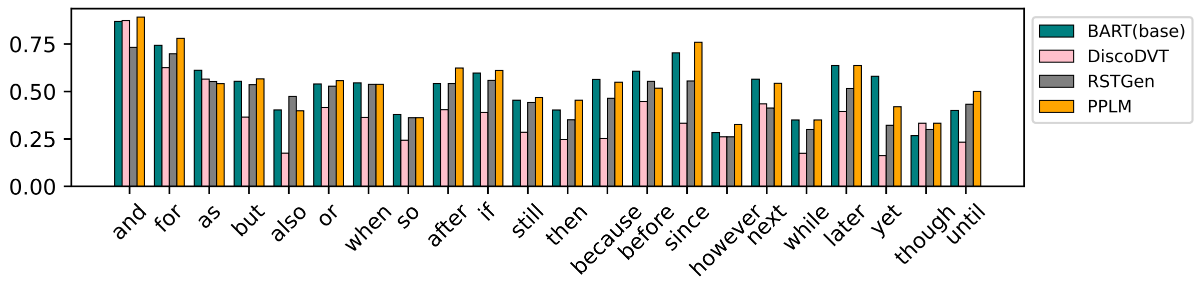

Figure 4: Experimental results for the recall of each discourse marker. We utilize discourse markers listed in Appendix A of the PDTB Annotation Manual (Prasad et al., 2007) We use only those discourse markers from the list that appear more than 30 times in the references.