# A Language Model-based Generative Classifier for Sentence-level Discourse Parsing

**Ying Zhang, Hidetaka Kamigaito and Manabu Okumura**
Tokyo Institute of Technology
{zhang,kamigaito,oku}@lr.pi.titech.ac.jp

## Abstract

Discourse segmentation and sentence-level discourse parsing play important roles for various NLP tasks to consider textual coherence. Despite recent achievements in both tasks, there is still room for improvement due to the scarcity of labeled data. To solve the problem, we propose a *language model-based generative classifier* (LMGC) for using more information from labels by treating the labels as an input while enhancing label representations by embedding descriptions for each label. Moreover, since this enables LMGC to make ready the representations for labels, unseen in the pre-training step, we can effectively use a pre-trained language model in LMGC. Experimental results on the RST-DT dataset show that our LMGC achieved the state-of-the-art $F_1$ score of 96.72 in discourse segmentation. It further achieved the state-of-the-art relation $F_1$ scores of 84.69 with gold EDU boundaries and 81.18 with automatically segmented boundaries, respectively, in sentence-level discourse parsing.

## 1 Introduction

Textual coherence is essential for writing a natural language text that is comprehensible to readers. To recognize the coherent structure of a natural language text, Rhetorical Structure Theory (RST) is applied to describe an internal discourse structure for the text as a constituent tree (Mann and Thompson, 1988). A discourse tree in RST consists of elementary discourse units (EDUs), spans that describe recursive connections between EDUs, and nuclearity and relation labels that describe relationships for each connection.

Figure 1 (a) shows an example RST discourse tree. A span including one or more EDUs is a node of the tree. Given two adjacent non-overlapping spans, their nuclearity can be either *nucleus* or *satellite*, denoted by N and S, where the *nucleus* represents a more salient or essential piece of information than the *satellite*. Furthermore, a relation

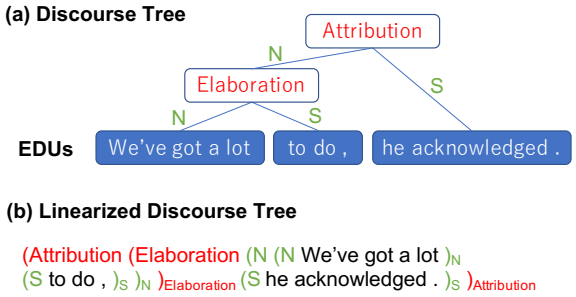



Figure 1: An example discourse tree structure.

label, such as *Attribution* and *Elaboration*, is used to describe the relation between the given spans (Mann and Thompson, 1988; Carlson and Marcu, 2001). To build such trees, RST parsing consists of discourse segmentation, a task to detect EDU boundaries in a given text, and discourse parsing, a task to link spans for detected EDUs.

In this paper, we focus on discourse segmentation and sentence-level discourse parsing, which are indispensable in RST parsing (Joty et al., 2013; Feng and Hirst, 2014a; Joty et al., 2015; Wang et al., 2017; Kobayashi et al., 2020) and are applicable to many downstream tasks, such as machine translation (Guzmán et al., 2014; Joty et al., 2017) and sentence compression (Sporleder and Lapata, 2005).

In discourse segmentation, Carlson et al. (2001) proposed a method for using lexical information and syntactic parsing results. Many researchers (Fisher and Roark, 2007; Xuan Bach et al., 2012; Feng and Hirst, 2014b) utilized these clues as features in a classifier although automatic parsing errors degraded segmentation performance. To avoid this problem, Wang et al. (2018b) used BiLSTM-CRF (Huang et al., 2015) to handle an input without these clues in an end-to-end manner. Lin et al. (2019) jointly performed discourse segmentation and sentence-level discourse parsing in their pointer-network-based model. They also intro-



*Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2432–2446
November 7–11, 2021. 

duced multi-task learning for both tasks and reported the state-of-the-art results for discourse segmentation and sentence-level discourse parsing in terms of $F_1$ scores. Despite these achievements, there is still room for improvement for both tasks due to the scarcity of labeled data. It is important to extract more potential information from the current dataset for further performance improvement.

Under this motivation, in this research, we propose a *language model-based generative classifier* (LMGC) as a reranker for both discourse segmentation and sentence-level discouse parsing. LMGC can jointly predict text and label probabilities by treating a text and labels as a single sequence, like Figure 1 (b). Therefore, different from conventional methods, LMGC can use more information from labels by treating the labels as an input. Furthermore, LMGC can enhance label representations by embedding descriptions of each label defined in the annotation manual (Carlson and Marcu, 2001), that allows us to use a pre-trained language model such as MPNet (Song et al., 2020) effectively, since we can already have the representations for labels, that were unseen in the pre-training step.

Experimental results on the RST-DT dataset (Carlson et al., 2002) show that LMGC can achieve the state-of-the-art scores in both discourse segmentation and sentence-level discourse parsing. LMGC utilizing our enhanced label embeddings achieves the best $F_1$ score of 96.72 in discourse segmentation. Furthermore, in sentence-level discourse parsing, LMGC utilizing our enhanced relation label embeddings achieves the best relation $F_1$ scores of 84.69 with gold EDU boundaries and 81.18 with automatically segmented boundaries, respectively.

## 2 Related Work

Discourse segmentation is a fundamental task for building an RST discourse tree from a text. Carlson et al. (2001) proposed a method for using lexical information and syntactic parsing results for detecting EDU boundaries in a sentence. Fisher and Roark (2007); Xuan Bach et al. (2012); Feng and Hirst (2014b) utilized these clues as features in a classifier, while Wang et al. (2018b) utilized BiLSTM-CRF (Huang et al., 2015) in an end-to-end manner to avoid performance degradation caused by syntactic parsing errors.

Sentence-level discourse parsing is also an important task for parsing an RST discourse tree, as used in many RST parsers (Joty et al., 2013; Feng and Hirst, 2014a; Joty et al., 2015; Wang et al., 2017; Kobayashi et al., 2020). Recently, Lin et al. (2019) tried to jointly perform discourse segmentation and sentence-level discourse parsing with pointer-networks and achieved the state-of-the-art $F_1$ scores in both discourse segmentation and sentence-level discourse parsing.

In spite of the performance improvement of these models, a restricted number of labeled RST discourse trees is still a problem. In the discourse segmentation and parsing tasks, most prior work is on the basis of discriminative models, which learn mapping from input texts to predicted labels. Thus, there still remains room for improving model performance by considering mapping from predictable labels to input texts to exploit more label information. To consider such information in a model, Mabona et al. (2019) introduced a generative model-based parser, RNNG (Dyer et al., 2016), to document-level RST discourse parsing. Different from our LMGC, this model unidirectionally predicts action sequences.

In this research, we model LMGC for the discourse segmentation and sentence-level discourse parsing tasks. LMGC utilizes a BERT-style bidirectional Transformer encoder (Devlin et al., 2019) to avoid prediction bias caused by using different decoding directions. Since LMGC is on the basis of generative models, it can jointly consider an input text and its predictable labels, and map the embeddings of both input tokens and labels onto the same space. Due to this characteristic, LMGC can effectively use the label information through constructing label embeddings from the description of a label definition (Carlson and Marcu, 2001). Furthermore, recent strong pre-trained models such as MPNet (Song et al., 2020) are available for any input tokens in LMGC.

## 3 Base Models

Our LMGC reranks the results from a conventional discourse segmenter and parser, which can be constructed as discriminative models. In this section, we explain these base models and introduce our mathematical notations.

### 3.1 Discourse Segmenter

In discourse segmentation, given an input text $\boldsymbol{x} = \{x_1, \cdots, x_n\}$, where $x_i$ is a word, a segmenter detects EDUs $\boldsymbol{e} = \{e_1, \cdots, e_m\}$ from $\boldsymbol{x}$. Since there is no overlap or gap between EDUs,

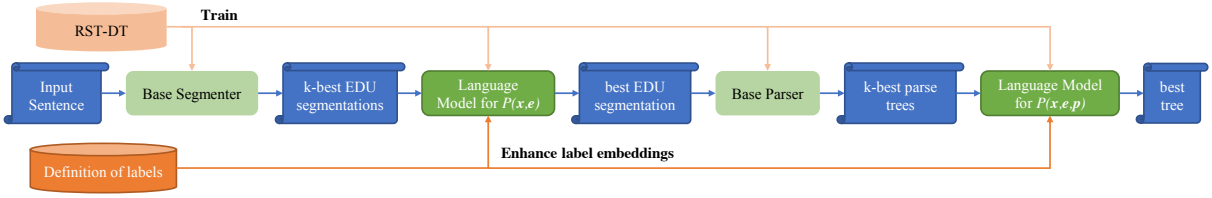



Figure 2: Overview of our Language Model-based Generative Classifier (LMGC).

discourse segmentation can be considered as a kind of sequential labeling task, which assigns labels $\boldsymbol{l} = \{l_1, \cdots, l_n\}$, where each $l_i \in \{0, 1\}$ indicates whether the word is the start of an EDU or not. By using a discriminative model, such as BiLSTM-CRF (Wang et al., 2018b) and pointer-networks (Lin et al., 2019), the probability of predicting EDUs from $\boldsymbol{x}$ can be $P(\boldsymbol{l}|\boldsymbol{x})$ or $P(\boldsymbol{e}|\boldsymbol{x})$. Because of its simple structure and extensibility, we choose BiLSTM-CRF as our base model for discourse segmentation. In BiLSTM-CRF, $P(\boldsymbol{l}|\boldsymbol{x})$ is formulated as follows:

$$P(\boldsymbol{l}|\boldsymbol{x}) = \frac{\prod_{t=1}^{n} \psi_t(l_t, l_{t-1}, h)}{\sum_{l' \in Y} \prod_{t=1}^{n} \psi_t(l_t', l_{t-1}', h)}, \quad (1)$$

where $\psi_t(l_t, l_{t-1}, h) = \exp(W^T h_t + b)$ is the potential function, $h_t$ is the hidden state at time step $t$, $W$ is a weight matrix, $b$ is a bias term, and $Y$ is the set of possible label sequences.

We inherit top-$k$ Viterbi results of BiLSTM-CRF, scored by Eq.(1), to our LMGC, as described in Section 4.

### 3.2 Discourse Parser

In discourse parsing, given an input text $\boldsymbol{x}$ and its EDUs $\boldsymbol{e}$, we can build a binary tree $\boldsymbol{p} = \{p_1, \cdots, p_{2n-1}\}$, where each node $p_i \in \boldsymbol{p}$ has three kinds of labels: span $s_i$, nuclearity $u_i$, and relation $r_i$. The sequences of span $\boldsymbol{s}$ and nuclearity $\boldsymbol{u}$ can be predicted simultaneously, as in 2-stage Parser (Wang et al., 2017), or span $\boldsymbol{s}$ can be predicted in advance for labeling nuclearity $\boldsymbol{u}$ and relation $\boldsymbol{r}$, as in pointer-networks (Lin et al., 2019) and span-based Parser (Kobayashi et al., 2020). Because of its better performance, we choose 2-stage Parser as our base model for sentence-level discourse parsing. 2-stage Parser extracts several features and does classification with SVMs in two stages. In the first stage, it identifies the span and nuclearity simultaneously to construct a tree based on the transition-based system with four types of actions: Shift, Reduce-NN, Reduce-NS, and Reduce-SN. In the second stage, for a given node $p_i$, $r_i$ is predicted as the relation between the left and right children nodes of $p_i$ by using features extracted from $p_i$ and its children nodes. In spite of its limited features, it achieves the best results compared with pointer-networks and span-based Parser. Since 2-stage Parser utilizes SVMs, we normalize the action scores and inherit top-$k$ beam search results of 2-stage Parser for LMGC to perform discourse parsing.

## 4 Language Model-based Generative Classifier (LMGC)

In this section, we introduce our generative classifier, LMGC, that utilizes a masked and permuted language model to compute sequence probabilities in both discourse segmentation and sentence-level discourse parsing tasks. More specifically, as we mention in Section 5, we can utilize our LMGC in three tasks, (a) discourse segmentation, (b) sentence-level discourse parsing with gold segmentation, and (c) sentence-level discourse parsing with automatic segmentation. Figure 2 shows the overview of our LMGC for the whole task (c). As shown in the figure, the prediction process in LMGC is the following. We assume that, in task (c), discourse segmentation and sentence-level discourse parsing are performed in a pipeline manner with models trained for tasks (a) and (b).

1. Predict top-$k_s$ EDU segmentations $\{\boldsymbol{e}_1, \cdots, \boldsymbol{e}_{k_s}\}$ from a given sentence $\boldsymbol{x}$ with the base discourse segmenter, described in Section 3.1.

2. Compute joint probability $P(\boldsymbol{x}, \boldsymbol{e_i})$ and select the best segmentation $\boldsymbol{e}$ from $\{\boldsymbol{e}_1, \cdots, \boldsymbol{e}_{k_s}\}$ with a language model, as we describe below.

3. Parse and rank top-$k_p$ trees $\{\boldsymbol{p}_1, \cdots, \boldsymbol{p}_{k_p}\}$ from $\boldsymbol{x}$ and best segmentation $\boldsymbol{e}$ with the base discourse parser, described in Section 3.2.

4. Compute joint probability $P(\boldsymbol{x}, \boldsymbol{e}, \boldsymbol{p_j})$ to select the best tree from $\{\boldsymbol{p}_1, \cdots, \boldsymbol{p}_{k_p}\}$ with a language model, as we describe below.

In task (a), we apply Step 2 to predict the best segmentation after Step 1. In task (b), we skip Steps 1 and 2, and apply just Steps 3 and 4 for gold segmentation to yield the best parse tree.

### 4.1 Tree Representations

To calculate joint probabilities for a discourse tree with a language model, we need to represent a tree as a linear form, like Figure 1 (b). Since there are several predictable label sets in discourse segmentation and parsing tasks, as shown in Figure 3, we prepare linearized forms for each label set.[1]

In discourse segmentation, we can consider joint probability $P(\boldsymbol{x}, \boldsymbol{e})$ for a sequence with inserting a symbol, [EDU], at an EDU boundary (Figure 3 (a)). In discourse parsing, a discourse tree is represented as a sequence with several kinds of label sets: span labels $\boldsymbol{s}$, nuclearity labels $\boldsymbol{u}$ including span labels, and relation labels $\boldsymbol{r}$ including span and nuclearity labels (Figures 3 (b)-(d)). To investigate the effectiveness of each label set in the reranking step, we consider $P(\boldsymbol{x}, \boldsymbol{e}, \boldsymbol{s})$, $P(\boldsymbol{x}, \boldsymbol{e}, \boldsymbol{u})$, and $P(\boldsymbol{x}, \boldsymbol{e}, \boldsymbol{r})$ for each label set to represent $P(\boldsymbol{x}, \boldsymbol{e}, \boldsymbol{p})$ in this paper. To build a sequence, we combine each label in a tree with brackets to imply the boundary for the label. For example, "(N" and ")$_{\mathrm{N}}$" stand for the start and end of a nucleus EDU. For a node $p_i$ of the tree, $r_i$ describes the relation between its children nodes, leading to $r_i$ of leaf nodes being "Null". When the child nodes of $p_i$ are *nucleus* and *satellite*, we assign label "Span" to the *nucleus* child node of $p_i$ and label $r_i$ to the *satellite* child node of $p_i$, respectively. When the child nodes of $p_i$ are both *nucleus*, we assign label $r_i$ to both child nodes of $p_i$.

For simpler illustration, in Figure 1 (b), we show the linearized discourse tree only with nuclearity and relation labels, since the nuclearity labels can also show span and EDU boundary labels. "Null" labels for leaf nodes are also omitted in the figure.

### 4.2 Joint Probabilities

To calculate joint probabilities in the last subsection with a language model, we consider probability $P(\boldsymbol{z})$ for a sequence $\boldsymbol{z} = (z_1, \cdots, z_a)$, which corresponds to the probabilities for the sequential representations $P(\boldsymbol{x}, \boldsymbol{e})$, $P(\boldsymbol{x}, \boldsymbol{e}, \boldsymbol{s})$, $P(\boldsymbol{x}, \boldsymbol{e}, \boldsymbol{u})$, and $P(\boldsymbol{x}, \boldsymbol{e}, \boldsymbol{r})$.

[1]Note that using just a raw s-expression-style tree of Figure 1 (b) in our language model cannot work because of its much more kinds of labels. We include the results with this type of tree in Appendix A.

According to Song et al. (2020), masked and permuted language modeling (MPNet) takes the advantages of both masked language modeling and permuted language modeling while overcoming their issues. Compared with Bert (Devlin et al., 2019) and XLNet (Yang et al., 2019), MPNet considered more information about tokens and positions, and achieved better results for several downsteam tasks (GLUE, SQuAD, etc). Taking into account its better performance, we choose pre-trained MPNet (Song et al., 2020) as our language model. Because considering all possible inter-dependence between $z_t$ is intractable, we follow the decomposition of pseudo-log-likelihood scores (PLL) (Salazar et al., 2020) in the model. Thus, we decompose and calculate logarithmic $P(\boldsymbol{z})$ as follows:

$$\begin{aligned} &\log P(\boldsymbol{z}; \theta) \qquad (2) \\ \approx &PLL(\boldsymbol{z}; \theta) = \sum_{t=1}^{a} \log P(z_t \,|\, z_{<t}, z_{>t}, M_t; \theta), \end{aligned}$$

where $z_{<t}$ is the first sub-sequence $(z_1, \cdots, z_{t-1})$ in $\boldsymbol{z}$ and $z_{>t}$ is the latter sub-sequence $(z_{t+1}, \cdots, z_a)$ in $\boldsymbol{z}$. $M_t$ denotes the mask token [MASK] at position $t$. $P(z_t \mid z_{<t}, z_{>t}, M_t; \theta)$ is computed by two-stream self-attention (Yang et al., 2019). In inference, we select $\boldsymbol{z}$ based on $\frac{1}{a} PLL(\boldsymbol{z}; \theta)$.

This model converts $\boldsymbol{z}$ into continuous vectors $\boldsymbol{w} = \{w_1, \cdots, w_a\}$ through the embedding layer. Multi-head attention layers further transform the vectors to predict each $z_t$ in the softmax layer.

Since pre-trained MPNet does not consider EDU, span, nuclearity, and relation labels in the pre-training step, we need to construct vectors $\boldsymbol{w}$ for these labels from the pre-trained parameters to enhance the prediction performance. We describe the details of this method in the next subsection.

### 4.3 Label Embeddings

In LMGC, we embed input text tokens and labels in the same vector space (Wang et al., 2018a) of the embedding layer. Under the setting, to deal with unseen labels in the pre-trained model, we compute the label embeddings by utilizing token embeddings in the pre-trained model.

We try to combine the input text with four kinds of labels, EDU, span, nuclearity, and relation labels, which were defined and clearly described in the annotation document (Carlson and Marcu, 2001) (See Appendix B for the descriptions). In taking into account the descriptions for the labels as additional

(a) Sentence with EDU boundary labels

$e_1$ _ [EDU] _ $e_2$ _ [EDU] _ $e_3$ _ [EDU]

(b) Sentence with span labels

(Span _ (Span _ $e_1$ _ )$_{\text{Span}}$ _ (Span _ $e_2$ _ )$_{\text{Span}}$ _ )$_{\text{Span}}$ _ (Span _ $e_3$ _ )$_{\text{Span}}$

(c) Sentence with nuclearity labels

(N _ (N _ $e_1$ _ )$_{\text{N}}$ _ (S _ $e_2$ _ )$_{\text{S}}$ _ )$_{\text{N}}$ _ (S _ $e_3$ _ )$_{\text{S}}$

(d) Sentence with relation labels

(Span _ (Span _ $e_1$ _ )$_{\text{Span}}$ _ (Elaboration _ $e_2$ _ )$_{\text{Elaboration}}$ _ )$_{\text{Span}}$ _ (Attribution _ $e_3$ _ )$_{\text{Attribution}}$

Figure 3: Example joint representations of an input text and labels for sentence *We've got a lot to do, he acknowledged.* $e_i$ represents the corresponding EDU, and "_" is whitespace.

information, we adopt two different methods, Average and Concatenate, for representing the label embeddings.

**Average**: We average the embeddings of tokens that appear in the definition of a label and assign the averaged embedding to the label.

**Concatenate**: We concatenate a label name with its definition and insert the concatenated text to the end of sequence $\boldsymbol{z}$,[2] so that the label embedding can be captured by self-attention mechanisms (Vaswani et al., 2017). Note that we do not try it in the parsing task, because the length of a sequence increases in proportion to the increase of the number of labels, that causes a shortage of memory space.

### 4.4 Objective Function

Because the search space for sequences of a text and its labels is exponentially large, instead of considering all possible sequences $Z(\boldsymbol{x})$ for $\boldsymbol{x}$, we assume $Z'(\boldsymbol{x})$ as a subset of sequences based on top-$k$ results from the base model. We denote $\boldsymbol{z}_g \in Z(\boldsymbol{x})$ as the correct label sequence of $\boldsymbol{x}$. To keep pre-trained information in MPNet, we continue masking and permutation for training model parameter $\theta$. Assuming that $O_a$ lists all permutations of set $\{1, 2, \cdots, a\}$, the number of elements in $O_a$ satisfies $\mid O_a \mid = a!$. For $\boldsymbol{z} \in Z'(\boldsymbol{x}) \cup \{\boldsymbol{z}_g\}$, we train the model parameter $\theta$ in LMGC by maximizing the following expectation over all permutations:

$$\mathbb{E}_{\boldsymbol{o} \in O_a} \sum_{t=c+1}^{a} [I_{\boldsymbol{z}} \log P(z_{o_t} \mid z_{o_{<t}}, M_{o_{>c}}; \theta) + (1 - I_{\boldsymbol{z}}) \log(1 - P(z_{o_t} \mid z_{o_{<t}}, M_{o_{>c}}; \theta))], \quad (3)$$

where $I_{\boldsymbol{z}}$ is the indicator function, defined as follows:

$$I_{\boldsymbol{z}} := \begin{cases} 1 & \text{if } \boldsymbol{z} = \boldsymbol{z}_g \\ 0 & \text{if } \boldsymbol{z} \neq \boldsymbol{z}_g \end{cases}. \quad (4)$$

$c$, denoting the number of non-predicted tokens $z_{o_{<=c}}$, is set manually. $o_{<t}$ denotes the first $t-1$ elements in $\boldsymbol{o}$. $M_{o_{>c}}$ denotes the mask tokens [MASK] at position $o_{>c}$. $P(z_{o_t} \mid z_{o_{<t}}, M_{o_{>c}}; \theta)$ is computed by two-stream self-attention (Yang et al., 2019).

## 5 Experiments

In this section, we present our experiments in three tasks, (a) discourse segmentation, (b) sentence-level discourse parsing with gold segmentation, and (c) sentence-level discourse parsing with automatic segmentation.

### 5.1 Experimental Settings

#### 5.1.1 Datasets

Following previous studies (Wang et al., 2017, 2018b; Lin et al., 2019), we used the RST Discourse Treebank (RST-DT) corpus (Carlson et al., 2002) as our dataset. This corpus contains 347 and 38 documents for training and test datasets, respectively. We divided the training dataset into two parts, following the module RSTFinder[3] (Heilman and Sagae, 2015), where 307 documents were used to train models and the remaining 40 documents were used as the validation dataset.

We split the documents into sentences while ignoring footnote sentences, as in Joty et al. (2012). There happens two possible problematic cases for the splitted sentences: (1) The sentence consists of exactly an EDU, and so it has no tree structure. (2) The tree structure of the sentence goes across to other sentences. Following the setting of Lin et al. (2019), we did not filter any sentences in task (a). In task (b), we filtered sentences of both cases. In task (c), we filtered sentences of case (2). Table 1 shows the number of available sentences for the three different tasks.

[2]Note that the concatenated text of the label name and its definition is not masked during training.

[3]https://github.com/EducationalTestingService/rstfinder

| Task | Train | Valid | Test |
|---|---|---|---|
| (a) Segmentation | 6,768 | 905 | 991 |
| (b) Parsing w/ gold segmentation | 4,524 | 636 | 602 |
| (c) Parsing w/ auto segmentation | - | 861 | 951 |

Table 1: The number of sentences for each task.

#### 5.1.2 Evaluation Metrics

In task (a), we evaluated the segmentation in micro-averaged precision, recall, and $F_1$ score with respect to the start position of each EDU. The position at the beginning of a sentence was ignored. In task (b), we evaluated the parsing in micro-averaged $F_1$ score with respect to span, nuclearity, and relation. In task (c) for parsing with automatic segmentation, we evaluated both the segmentation and parsing in micro-averaged $F_1$ score.

We used the paired bootstrap resampling (Koehn, 2004) for the significance test in all tasks when comparing two systems.

#### 5.1.3 Compared Methods

As our proposed methods, we used LMGC$_{\boldsymbol{e}}$, LMGC$_{\boldsymbol{s}}$, LMGC$_{\boldsymbol{u}}$, and LMGC$_{\boldsymbol{r}}$, which respectively model probability $P(\boldsymbol{x}, \boldsymbol{e})$, $P(\boldsymbol{x}, \boldsymbol{e}, \boldsymbol{s})$, $P(\boldsymbol{x}, \boldsymbol{e}, \boldsymbol{u})$, and $P(\boldsymbol{x}, \boldsymbol{e}, \boldsymbol{r})$ with initialized label embeddings. We represent LMGC with Average and Concatenate label embeddings as Enhance and Extend, respectively.

We used the base discourse segmenter and parser described in Section 3 as our baseline. We reproduced the base discourse segmenter BiLSTM-CRF[4] (Wang et al., 2018b). Because BiLSTM-CRF adopted the hidden states of ELMo (Peters et al., 2018) as word embeddings, we also tried the last hidden state of MPNet as the word embeddings for BiLSTM-CRF for fairness. We retrained the segmenter in five runs, and the experimental results are showed in Appendix C. The publicly shared BiLSTM-CRF by Wang et al. (2018b) is our base segmenter in the following experiments.

As for the base parser, we retrained two models, 2-stage Parser[5] (Wang et al., 2017) and span-based Parser[6] (Kobayashi et al., 2020). Different from the setting of Lin et al. (2019), we retrained 2-stage Parser in the sentence-level rather than in the document-level. Since the experimental results show our retrained 2-stage Parser achieved the highest $F_1$ scores among several parsers (See Appendix C), we selected it as our base parser in the following experiments.

Furthermore, for comparing LMGC with an unidirectional generative model (Mabona et al., 2019), we constructed another baseline method which utilizes a GPT-2 (Radford et al., 2019) based reranker. This method follows an unidirectional language model-based generative parser (Choe and Charniak, 2016), and considers top-$k$ results from the base model by an add-1 version of infinilog loss (Ding et al., 2020) during training. We denote this baseline as GPT2LM hereafter. GPT2LM models $P(\boldsymbol{x}, \boldsymbol{e})$ for task (a) and $P(\boldsymbol{x}, \boldsymbol{e}, \boldsymbol{r})$ for tasks (b) and (c), respectively. Both LMGC and GPT2LM are the ensemble of 5 models with different random seeds. See Appendix D for a complete list of hyperparameter settings.

#### 5.1.4 Number of Candidates

As described in Section 4, LMGC requires parameters $k_s$ and $k_p$ for the number of candidates in the steps for different tasks. We tuned $k_s$ and $k_p$ based on the performance on the validation dataset.[7]

In task (a), $k_s$ was set to 20 and 5 for training and prediction, respectively. In task (b), $k_p$ was set to 20 and 5 for training and prediction, respectively. In task(c), $k_s$ and $k_p$ were both set to 5 for prediction. The set of parameters was similarly tuned for GPT2LM on the validation dataset. We list all of them in Appendix E.

### 5.2 Results

#### 5.2.1 Discourse Segmentation

Table 2 shows the experimental results for the discourse segmentation task. Oracle indicates the upper bound score that can be achieved with candidates generated by the base model. To compute the Oracle score, if the candidades by the base model include the correct answer, we assume the prediction is correct.

LMGC$_e$ significantly outperformed GPT2LM$_e$.[8] We think the reason is similar to what Zhu et al. (2020) reported: BERT-based bidirectional Transformer encoders encode more rhetorical features than GPT2-based unidirectional Transformer en-

[4] https://github.com/PKU-TANGENT/NeuralEDUSeg

[5] https://github.com/yizhongw/StageDP

[6] https://github.com/nttcslab-nlp/Top-Down-RST-Parser

[7] Note that we should separately tune the number of candidates for training and prediction stages because LMGC utilizes Eq.(2) for prediction and Eq.(3) for training, respectively.

[8] We chose GPT2LM$_e$ for the significance test because we had only reported scores for the pointer-networks.

| Model | Precision | Recall | $F_1$ |
|---|---|---|---|
| Oracle | 97.73 | 98.67 | 98.20 |
| Pointer-networks* | 93.34 | 97.88 | 95.55 |
| Base segmenter | 92.22 | 95.35 | 93.76 |
| GPT2LM$_e$ | 94.05 | 95.72 | 94.88 |
| LMGC$_e$ | 95.31 | 97.56 | 96.43† |
| Enhance$_e$ | **95.54** | **97.93** | **96.72**† |
| Extend$_e$ | 95.05 | 97.86 | 96.44† |

Table 2: Results for the discourse segmentation task. * indicates the reported score by Lin et al. (2019). The best score in each metric among the models is indicated in **bold**. † indicates that the score is significantly superior to GPT2LM with a p-value < 0.01.

| Model | Span | Nuclearity | Relation |
|---|---|---|---|
| Oracle | 98.67 | 95.88 | 90.07 |
| Pointer-networks* | 97.44 | 91.34 | 81.70 |
| Base parser | 97.92 | 92.07 | 82.06 |
| GPT2LM$_r$ | 96.35 | 88.11 | 77.86 |
| LMGC$_s$ | 98.23‡ | 92.31 | 82.22 |
| Enhance$_s$ | 98.27‡ | 92.39 | 82.42 |
| LMGC$_u$ | **98.31**‡ | **94.00**† | 83.63† |
| Enhance$_u$ | **98.31**† | 93.88† | 83.56† |
| LMGC$_r$ | 98.00 | 93.09† | 83.99† |
| Enhance$_r$ | 98.12 | 93.13† | **84.69**† |

Table 3: Results for the sentence-level discourse parsing task with gold segmentation. * indicates the reported score by Lin et al. (2019). The best score in each metric among the models is indicated in **bold**. † and ‡ indicate that the score is significantly superior to the base parser with a p-value < 0.01 and < 0.05, respectively.

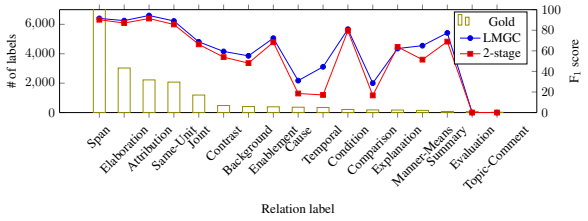



Figure 4: Performance of 2-stage parser and Enhance$_r$ in the sentence-level discourse parsing task with gold segmentation. The hollow bar denotes the number of different gold labels in the training dataset. Blue and red lines indicate the $F_1$ scores of Enhance$_r$ and 2-stage parser, respectively, for each relation label.

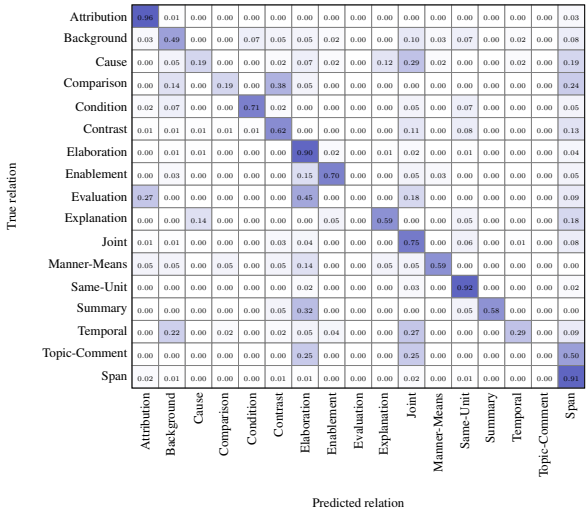



Figure 5: Confusion matrix for Enhance$_r$ in the sentence-level discourse parsing task with gold segmentation. We show the ratio of the number of instances with predicted labels (for a column) to the number of instances with gold labels (for a row) in the corresponding cell.

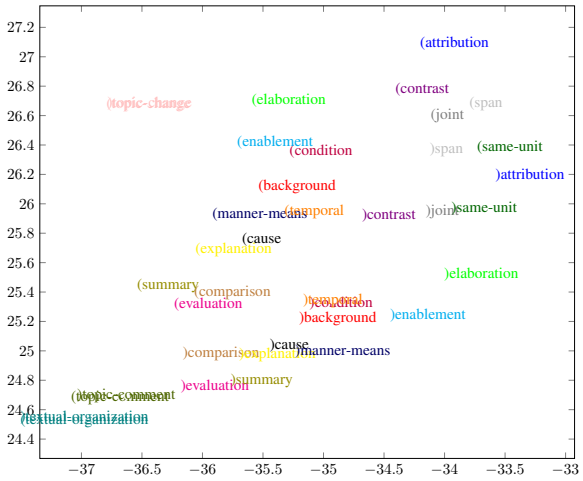
coders. Using Average label embeddings is more helpful than using Concatenate label embeddings for LMGC$_e$. Enhance$_e$ achieved the state-of-the-art $F_1$ score of 96.72, which outperformed both the base segmenter and the pointer-networks.

### 5.2.2 Sentence-level Discourse Parsing

**Gold Segmentation**: Table 3, Figures 4 and 5 show the experimental results for the sentence-level discourse parsing task with gold segmentation. In Table 3, LMGC$_u$ achieved the highest span and nuclearity $F_1$ scores of 98.31 and 94.00, respectively. Enhance$_r$ achieved the state-of-the-art relation $F_1$ score of 84.69, which is significantly superior to the base parser. Although using Average label embeddings improved LMGC$_r$, it can provide no or only limited improvement for LMGC$_u$ and LMGC$_s$. We guess that this difference is caused by the number of different kinds of labels in span, nuclearity, and relation. The performance of GPT2LM$_r$ is even worse than the base parser. We think this is because we added the relation labels to the vocabulary of GPT-2 and resized the pre-trained word embeddings.

Figure 4 shows the comparison between the base parser and Enhance$_r$ with respect to each ralation label. In most relation labels, Enhance$_r$ outperformed 2-stage Parser except for the labels *Explanation*, *Evaluation*, and *Topic-Comment*. 2-stage Parser achieved the $F_1$ score of 17.14 for label *Temporal* while Enhance$_r$ achieved the $F_1$ score of 44.44 by reranking the parsing results from 2-stage Parser. Such great improvement with Enhance$_r$ can also be found for labels such as *Contrast*, *Back-*

(a) LMGC$_r$




(b) Enhance$_r$

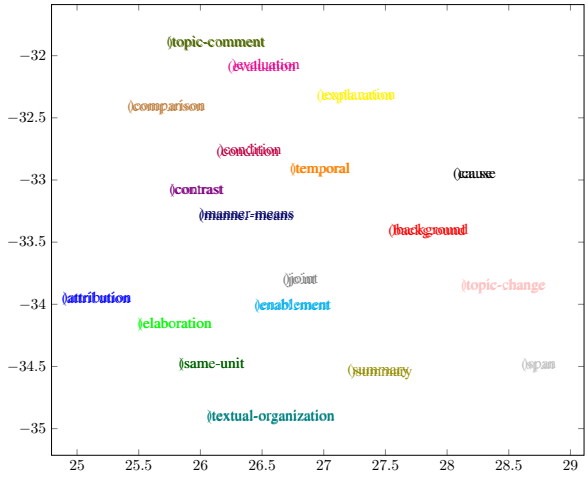



Figure 6: t-SNE plot of relation label embeddings trained in LMGC$_r$ and Enhance$_r$.

*ground*, and *Cause*. Obviously, Enhance$_r$ tends to improve the performance for labels whose training data is limited.

Figure 5 shows a confusion matrix of Enhance$_r$ for each relation label. It shows that the relation labels *Comparison*, *Cause*, and *Temporal* were often predicted wrongly as *Contrast*, *Joint*, and *Joint* or *Background*, respectively, by Enhance$_r$, even though these labels have at least 100 training data. We guess this might be due to some similarities between those labels.

By using the t-SNE plot (Van der Maaten and Hinton, 2008), we visualize the trained relation label embeddings of LMGC$_r$ and Enhance$_r$. Figures 6a and 6b show the results. Figure 6a shows a clearer diagonal that divides labels with parenthesis

| Model | Seg | Parse | | |
|---|---|---|---|---|
| | | Span | Nuclearity | Relation |
| Pointer-networks* | - | 91.75 | 86.38 | 77.52 |
| Oracle$_{seg}$ | 98.24 | - | - | - |
| Base segmenter | 93.92 | - | - | - |
| GPT2LM$_e$ | 95.03 | - | - | - |
| LMGC$_e$ | 96.51 | - | - | - |
| Enhance$_e$ | **96.79** | - | - | - |
| Extend$_e$ | 96.48 | - | - | - |
| Oracle | - | 93.95 | 91.25 | 85.93 |
| Base parser | - | 93.53 | 88.08 | 78.75 |
| GPT2LM$_r$ | - | 92.02 | 84.20 | 74.49 |
| LMGC$_s$ | - | 93.96‡ | 88.46 | 79.25 |
| Enhance$_s$ | - | **94.00**† | 88.50 | 79.33 |
| LMGC$_u$ | - | 93.96† | **89.90**† | 80.33† |
| Enhance$_u$ | - | 93.92‡ | 89.74† | 80.22† |
| LMGC$_r$ | - | 93.65 | 89.08† | 80.57† |
| Enhance$_r$ | - | 93.73 | 89.16† | **81.18**† |

Table 4: Results for the sentence-level discourse parsing task with automatic segmentation. * indicates the reported score by Lin et al. (2019). The best score in each metric among the models for each block is indicated in **bold**. We used the discourse segmentation results of Enhance$_e$ as the input of the discourse parsing stage for all models, for fair comparison of sentence-level discourse parsing. † and ‡ indicate that the score is significantly superior to the base parser with a p-value $< 0.01$ and $< 0.05$, respectively.

"(" from the ones with ")", while Figure 6b shows more distinct divisions between labels.

**Automatic Segmentation**: Table 4 shows the experimental results for the sentence-level discourse parsing task with automatic segmentation. The second and third blocks in the table show the results for the first and second stages, discourse segmentation and sentence-level discourse parsing, respectively.[9]

Enhance$_r$ achieved the highest relation $F_1$ score of 81.18, which is a significant improvement of 2.43 points compared to the base parser. Enhance$_s$ and LMGC$_u$ achieved the highest span and nuclearity $F_1$ scores of 94.00 and 89.90, respectively. Since LMGC$_*$ and Enhance$_*$ were the models trained in task (b), and Enhance$_e$ achieved the $F_1$ score of 96.79 in discourse segmentation, it is not surprising to find that the tendency of those results is similar to that in sentence-level discourse parsing with gold segmentation.

## 6 Conclusion

In this research, we proposed a *language model-based generative classifier*, LMGC. Given the top-

[9]Note that $F_1$ scores for discourse segmentation in the second block are not the same as in Table 2 due to the different test dataset.

$k$ discourse segmentations or parsings from the base model, as a reranker, LMGC achieved the state-of-the-art performances in both discourse segmentation and sentence-level discourse parsing. The experimental results also showed the potential of constructing label embeddings from token embeddings by using label descriptions in the manual. In the future, we plan to apply LMGC to other diverse classification tasks.

## References


Lynn Carlson and Daniel Marcu. 2001. Discourse tagging reference manual. *ISI Technical Report ISI-TR-545*.

Lynn Carlson, Daniel Marcu, and Ellen Okurowski Mary. 2002. Rst discourse treebank ldc2002t07. *Philadelphia:Linguistic Data Consortium*.

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*.

Do Kook Choe and Eugene Charniak. 2016. Parsing as language modeling. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2331–2336, Austin, Texas. Association for Computational Linguistics.

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.

Xiaoan Ding, Tianyu Liu, Baobao Chang, Zhifang Sui, and Kevin Gimpel. 2020. Discriminatively-Tuned Generative Classifiers for Robust Natural Language Inference. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8189–8202, Online. Association for Computational Linguistics.

Chris Dyer, Adhiguna Kuncoro, Miguel Ballesteros, and Noah A. Smith. 2016. Recurrent neural network grammars. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 199–209, San Diego, California. Association for Computational Linguistics.

Vanessa Wei Feng and Graeme Hirst. 2014a. 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 (Volume 1: Long Papers)*, pages 511–521, Baltimore, Maryland. Association for Computational Linguistics.

Vanessa Wei Feng and Graeme Hirst. 2014b. Two-pass discourse segmentation with pairing and global features.

Seeger Fisher and Brian Roark. 2007. The utility of parse-derived features for automatic discourse segmentation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 488–495, Prague, Czech Republic. Association for Computational Linguistics.

Francisco Guzmán, Shafiq Joty, Lluís Màrquez, and Preslav Nakov. 2014. Using discourse structure improves machine translation evaluation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 687–698, Baltimore, Maryland. Association for Computational Linguistics.

Michael Heilman and Kenji Sagae. 2015. Fast rhetorical structure theory discourse parsing. *arXiv preprint arXiv:1505.02425*.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional lstm-crf models for sequence tagging.

Shafiq Joty, Giuseppe Carenini, and Raymond 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*, pages 904–915, Jeju Island, Korea. Association for Computational Linguistics.

Shafiq Joty, Giuseppe Carenini, Raymond 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 (Volume 1: Long Papers)*, pages 486–496, Sofia, Bulgaria. Association for Computational Linguistics.

Shafiq Joty, Giuseppe Carenini, and Raymond T. Ng. 2015. CODRA: A novel discriminative framework for rhetorical analysis. *Computational Linguistics*, 41(3):385–435.

Shafiq Joty, Francisco Guzmán, Lluís Màrquez, and Preslav Nakov. 2017. Discourse structure in machine translation evaluation. *Computational Linguistics*, 43(4):683–722.

Dan Jurafsky. 2000. *Speech & language processing*. Pearson Education India.

Naoki Kobayashi, Tsutomu Hirao, Hidetaka Kamigaito, Manabu Okumura, and Masaaki Nagata. 2020. Top-down rst parsing utilizing granularity levels in documents. volume 34, pages 8099–8106.

Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 388–395, Barcelona, Spain. Association for Computational Linguistics.

Xiang Lin, Shafiq Joty, Prathyusha Jwalapuram, and M Saiful Bari. 2019. A unified linear-time framework for sentence-level discourse parsing. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4190–4200, Florence, Italy. Association for Computational Linguistics.

Amandla Mabona, Laura Rimell, Stephen Clark, and Andreas Vlachos. 2019. Neural generative rhetorical structure 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)*, pages 2284–2295, Hong Kong, China. Association for Computational Linguistics.

William C Mann and Sandra A Thompson. 1988. Rhetorical structure theory: Toward a functional theory of text organization.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53, Minneapolis, Minnesota. Association for Computational Linguistics.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Julian Salazar, Davis Liang, Toan Q. Nguyen, and Katrin Kirchhoff. 2020. Masked language model scoring. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2699–2712, Online. Association for Computational Linguistics.

Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. 2018. Self-attention with relative position representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 464–468, New Orleans, Louisiana. Association for Computational Linguistics.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2020. Mpnet: Masked and permuted pre-training for language understanding. *arXiv preprint arXiv:2004.09297*.

Caroline Sporleder and Mirella Lapata. 2005. Discourse chunking and its application to sentence compression. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 257–264, Vancouver, British Columbia, Canada. Association for Computational Linguistics.

Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research*, 9(11).

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NIPS*.

Guoyin Wang, Chunyuan Li, Wenlin Wang, Yizhe Zhang, Dinghan Shen, Xinyuan Zhang, Ricardo Henao, and Lawrence Carin. 2018a. Joint embedding of words and labels for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2321–2331, Melbourne, Australia. Association for Computational Linguistics.

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 (Volume 2: Short Papers)*, pages 184–188, Vancouver, Canada. Association for Computational Linguistics.

Yizhong Wang, Sujian Li, and Jingfeng Yang. 2018b. 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.

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

Ngo Xuan Bach, Nguyen Le Minh, and Akira Shimazu. 2012. A reranking model for discourse segmentation using subtree features. In *Proceedings of the 13th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 160–168, Seoul, South Korea. Association for Computational Linguistics.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *arXiv preprint arXiv:1906.08237*.

Zining Zhu, Chuer Pan, Mohamed Abdalla, and Frank Rudzicz. 2020. Examining the rhetorical capacities of neural language models. In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 16–32.

## A Experimental Results of LMGC with Tree

Since the raw s-expression-style tree is longer than our joint representations with span, nuclearity and relation, we transformed the raw tree into a sequence as Figure 7 shows, where the nuclearity and relation labels are connected together by the colons. To construct the label embedding for $P(\boldsymbol{x}, \boldsymbol{e}, \boldsymbol{p})$, we combined the descriptions of the nuclearity and relation (see descriptions in Appendix B), and assigned the combination to the corresponding node. For example, the description of "(Attribution:S" is *the start of a supporting or background piece of information attribution, attribution represents both direct and indirect instances of reported speech*.

(Span:N _ (Span:N _ $e_1$ _ )$_{\text{Span:N}}$ _ (Elaboration:S _ $e_2$ _ )$_{\text{Elaboration:S}}$ _ )$_{\text{Span:N}}$ _ (Attribution:S _ $e_3$ _ )$_{\text{Attribution:S}}$

Figure 7: Example joint representation of an input text with all tree labels for sentence *We've got a lot to do, he acknowledged.* $e_i$ represents the corresponding EDU, and "_" is whitespace.

LMGC$_{\boldsymbol{p}}$ models the joint probability $P(\boldsymbol{x}, \boldsymbol{e}, \boldsymbol{p})$ with initialized label embedding. The experimental results of LMGC$_{\boldsymbol{p}}$ and Enhance$_{\boldsymbol{p}}$ for the sentence-level discourse parsing task with gold segmentation are showed in Table 5. LMGC$_{\boldsymbol{p}}$ and Enhance$_{\boldsymbol{p}}$ are the ensemble of 5 models with different random seed, although the training loss of Enhance$_{\boldsymbol{p}}$ in 2 of 5 models did not decrease.

| Model | Span | Nuclearity | Relation |
|---|---|---|---|
| LMGC$_{\boldsymbol{p}}$ | 97.84 | 92.90 | 84.11 |
| Enhance$_{\boldsymbol{p}}$ | 98.04 | 92.74 | 84.18 |

Table 5: Performances of LMGC$_{\boldsymbol{p}}$ and Enhance$_{\boldsymbol{p}}$ in the sentence-level discourse parsing task with gold segmentation.

## B Label Descriptions

We list our extracted label descriptions from Carlson and Marcu (2001) in Table 6. For parsing symbols with brackets "(" and ")" like "(N" and ")$_{\text{N}}$", we inserted the position phrase, *the start of* and *the end of*, to the beginning of their label definitions. So the description of ")$_{\text{N}}$" is *the end of a more salient or essential piece of information*.

## C Experiment Results of Reproduced Base Model

Table 7 shows the experimental results of BiLSTM-CRF in discourse segmentation, where the results of our reproduced BiLSTM-CRF are averaged in five runs. Table 8 shows the experimental results of different parsers in the sentence-level discourse parsing task with gold segmentation.

## D Hyperparmeters

For LMGC, we used the source code shared in the public github[10] of Song et al. (2020). We used the uploaded pre-trained MPNet and same setup as illustrated in Table 9. 15% tokens as the predicted tokens were masked by replacement strategy 8:1:1. Relative positional embedding mechanism (Shaw et al., 2018) was utilized. Since the vocab we used is same as the one of BERT (Devlin et al., 2019), we used the symbol [SEP] to represent [EDU] and symbol [unused#] starting from 0 to represent parsing labels such as "(N" and "(Attribution".

For GPT2LM, we used the source code shared in the public github[11] (Ott et al., 2019). Following the steps in Choe and Charniak (2016), we utilized Eq (5) (Jurafsky, 2000) to compute the joint distribution,

$$\begin{aligned} P(\boldsymbol{x}, \boldsymbol{y}) = P(\boldsymbol{z}) &= P(z_1, \ldots, z_a) \\ &= \prod_{t=1}^{a} P(z_t | z_1, \ldots, z_{t-1}), \end{aligned} \quad (5)$$

where $P(z_t|z_1, \ldots, z_{t-1})$ was computed by GPT-2 (Radford et al., 2019). And in inference, we selected $\boldsymbol{z}$ based on $\frac{1}{a}\log P(\boldsymbol{z})$. An add-1 version of infinilog loss (Ding et al., 2020) was utilized for training GPT2LM as follows:

$$-\log f(\boldsymbol{z}) + \log[1 + \sum_{\boldsymbol{z}' \in Z'(\boldsymbol{x}), \boldsymbol{z}' \neq \boldsymbol{z}} f(\boldsymbol{z}')], \quad (6)$$

[10] https://github.com/microsoft/MPNet

[11] https://github.com/pytorch/fairseq/tree/master/fairseq/models/huggingface

| Label | Definition |
|---|---|
| [EOS] | elementary discourse units are the minimal building blocks of a discourse tree |
| Span | span |
| Nucleus | a more salient or essential piece of information |
| Satellite | a supporting or background piece of information |
| Attribution | attribution, attribution represents both direct and indirect instances of reported speech |
| Background | background or circumstance |
| Cause | cause or result |
| Comparison | comparison, preference, analogy or proportion |
| Condition | condition, hypothetical, contingency or otherwise |
| Contrast | contrast relation, spans contrast with each other along some dimension. Typically, it includes a contrastive discourse cue, such as but, however, while. |
| Elaboration | elaboration, elaboration provides specific information or details to help define a very general concept |
| Enablement | enablement, enablement presentes action to increase the chances of the unrealized situation being realized. |
| Evaluation | evaluation, interpretation, conclusion or comment |
| Explanation | evidence, explanation or reason |
| Joint | list, list contains some sort of parallel structure or similar fashion between the units |
| Manner-Means | explaining or specifying a method , mechanism , instrument , channel or conduit for accomplishing some goal |
| Topic-Comment | problem solution, question answer, statement response, topic comment or rhetorical question |
| Summary | summary or restatement |
| Temporal | situations with temporal order, before, after or at the same time |
| Topic change | topic change |
| Textual-organization | links that are marked by schemata labels |
| Same-unit | links between two non-adjacent parts when separated by an intervening relative clause or parenthetical |

Table 6: Extracted label definitions.

| Model | Precision | Recall | $F_1$ |
|---|---|---|---|
| Reported* | 92.04 | 94.41 | 93.21 |
| Shared | 92.22 | 95.35 | 93.76 |
| Reproduced (ELMo) | **93.16** | **96.26** | **94.68** |
| Reproduced (MPNet) | 92.84 | 95.63 | 94.21 |

Table 7: Performances of BiLSTM-CRF (Wang et al., 2018b) in the discourse segmentation task. The best score in each metric among the models is indicated in **bold**. * indicates the reported score by Lin et al. (2019). Shared is the publicly shared model by Wang et al. (2018b). Reproduced (ELMo) and Reproduced (MPNet) are our reproduced models with different word embeddings.

| Model | Span | Nuclearity | Relation |
|---|---|---|---|
| 2-Stage Parser* | 95.60 | 87.80 | 77.60 |
| Pointer-networks* | 97.44 | 91.34 | 81.70 |
| Span-based Parser | 96.67 | 90.23 | 74.76 |
| 2-Stage Parser | **97.92** | **92.07** | **82.06** |

Table 8: Performance of retrained parsers in the sentence-level discourse parsing task with gold segmentation. The best score in each metric among the models is indicated in **bold**. * indicates the reported score by Lin et al. (2019).

where

$$f(\boldsymbol{z}) = \frac{\exp(\frac{1}{a}\log P(\boldsymbol{z}))}{\sum_{\boldsymbol{z}' \in Z'(\boldsymbol{x})} \exp(\frac{1}{a'}\log P(\boldsymbol{z}'))}. \quad (7)$$

We used the uploaded pretrained "gpt2" model (Wolf et al., 2020) and same setup as illustrated in Table 10. We used symbol "=====" in vocab to represent the symbol [EDU]. Because the vocab of GPT-2 has no available symbol for representing an unseen symbol, we added <pad> and our relation symbols to the vocab of GPT-2 and resized the pre-trained word embeddings.

## E Setting of Candidates

Table 11 shows the setting of candidates for different tasks. As described in Section 4.4, we do data augmentation by using additional top-$k$ results generated by a base method, a larger $k$ during training is expected to bring more promotion for LMGC. However, a larger $k$ during prediction step introduces more candidates and may make the prediction more difficult. Taking this into consideration, we tuned $k_s$ and $k_p$ for training and prediction separately based on the performance on the validation dataset.

| Hyperparameter | Value |
|---|---|
| Optimizer | adam |
| Adam $\beta_1$ | 0.9 |
| Adam $\beta_2$ | 0.98 |
| Adam $\epsilon$ | $1e$ - 6 |
| weight decay | 0.01 |
| Learning rate | 0.00009 |
| Batch size | 8192 tokens |
| Warm up steps | 2.4 epoch |
| Epoch | 30 |
| Attention layer | 12 |
| Attention head | 12 |
| dropout | 0.1 |
| attention dropout | 0.1 |
| Hidden size | 768 |
| Vocab size | 30527 |
| Tokenizer | Byte pair encoder |
| Max sentence length | 512 |

Table 9: List of used hyperparameters for LMGC.

| Hyperparameter | Value |
|---|---|
| Optimizer | adam |
| Adam $\beta_1$ | 0.9 |
| Adam $\beta_2$ | 0.98 |
| Adam $\epsilon$ | $1e$ - 6 |
| weight decay | 0.01 |
| Learning rate | 0.0001 |
| Batch size | 512 gold tokens + candidate tokens |
| Warm up steps | 2.4 epoch |
| Epoch | 30 |
| Attention layer | 12 |
| Attention head | 12 |
| dropout | 0.1 |
| attention dropout | 0.1 |
| Hidden size | 768 |
| Vocab size | 50257+ added tokens |
| Tokenizer | Byte pair encoder |
| Max sentence length | 512 |

Table 10: List of used hyperparameters for GPT2LM.

In task (a), we used the Viterbi-topk algorithm for the base segmenter to select top-$k_s$ segmentations. We tuned $k_s \in \{0, 10, 20\}$ for training while $k_s$ for prediction was fixed as 5. Note that we used only gold segmentations for training when $k_s$ was set to 0. Table 12 shows the experimental results, where both LMGC$_e$ and GP2TLM$_e$ are the ensemble of 5 models. Then we tuned $k_s \in \{5, 10, 20\}$ for prediction by using the LMGC$_e$ and GP2TLM$_e$ trained with top-20 candidates, Table 13 shows the results.

In task (b), we utilized beam search in each stage

| Task | Data | Segmentation | Parsing | | | # of data |
|---|---|---|---|---|---|---|
| | | $k_s$ | 1st stage | 2rd stage | $k_p$ | |
| (a) | Training | 20 | - | - | - | 140924 |
| | Prediction | 5 | - | - | - | - |
| (b) | $\text{Training}_{\text{w/ span or nuclearity}}$ | - | 20 | 1 | 20 | 60742 |
| | $\text{Training}_{\text{w/ relation or all}}$ | - | 3 | 7 | 20 | 95004 |
| | Prediction | - | 5 | 5 | 5 | - |
| (c) | Prediction | 5 | 5 | 5 | 5 | - |

Table 11: Setting of top candidates for different tasks. The Prediction data denotes the validation and test dataset.

of the base parser and after two stages we computed the perplexity to keep top-$k_p$ parsings. We tuned $k_p \in \{0, 10, 20\}$ for training while $k_p$ for prediction was fixed as 5. Note that we used only gold parsings for training when $k_p$ was set to 0. Table 14 shows the experimental results, where both $\text{LMGC}_r$ and $\text{GPT2LM}_r$ are the ensemble of 5 models. Then we tuned $k_p \in \{5, 10, 20\}$ for prediction by using the $\text{LMGC}_r$ and $\text{GPT2LM}_r$ trained with top-20 candidates, Table 15 shows the results.

In task (c), same as in task (a), we tuned $k_s \in \{5, 10, 20\}$ for predicting discourse segmentation by using the $\text{LMGC}_e$ and $\text{GP2TLM}_e$ trained with top-20 candidates for task (a), Table 16 shows the result. We utilized $\text{LMGC}_e$ to select the best segmentation from top-5 segmentations for following discourse parsing. Then same as in task (b), we tuned $k_p \in \{5, 10, 20\}$ for predicting discourse parsing by using the $\text{LMGC}_r$ and $\text{GPT2LM}_r$ trained with top-20 candidates for task (b), Table 17 shows the result.

In tasks (b) and (c), $\text{LMGC}_{\boldsymbol{s}}$ and $\text{Enhance}_{\boldsymbol{s}}$ cannot distinguish the candidates with the same span labels but different nulearity or relation labels, $\text{LMGC}_{\boldsymbol{u}}$ and $\text{Enhance}_{\boldsymbol{u}}$ cannot distinguish the candidates with the same nulearity labels but different relation labels. Under this condition, the indistinguishable parsings would be ranked by the base parser. And in task (b), for training data with span or nuclearity labels, we used the beam sizes 20 and 1 in the first and second stages of the base parser, respectively.

| Model | $k_s$ for training | Precision | Recall | $\text{F}_1$ |
|---|---|---|---|---|
| $\text{LMGC}_e$ | 0 | 87.76 | 95.72 | 91.57 |
| | 10 | 97.67 | 97.73 | 97.70 |
| | 20 | **97.99** | **97.86** | **97.92** |
| $\text{GPT2LM}_e$ | 0 | 81.72 | **96.18** | 88.36 |
| | 10 | 96.67 | 96.05 | 96.36 |
| | 20 | **96.93** | 96.05 | **96.48** |

Table 12: Results of tuning $k_s$ for training in task (a). The best score in each metric among different $k_s$ for training is indicated in **bold**.

| Model | $k_s$ for prediction | Precision | Recall | $\text{F}_1$ |
|---|---|---|---|---|
| Oracle | 5 | 99.94 | 99.68 | 99.81 |
| | 10 | 99.94 | 99.68 | 99.81 |
| | 20 | 99.94 | 99.68 | 99.81 |
| $\text{LMGC}_e$ | 5 | **97.99** | **97.86** | **97.92** |
| | 10 | 97.47 | 97.54 | 97.51 |
| | 20 | 97.41 | 97.60 | 97.51 |
| $\text{GPT2LM}_e$ | 5 | **96.93** | **96.05** | **96.48** |
| | 10 | 96.47 | 95.59 | 96.03 |
| | 20 | 95.76 | 95.14 | 95.45 |

Table 13: Results of tuning $k_s$ for prediction in task (a). The best score in each metric among different $k_s$ for prediction is indicated in **bold**.

| Model | $k_p$ for training | Span | Nuclearity | Relation |
|---|---|---|---|---|
| $\text{LMGC}_r$ | 0 | 97.25 | 92.21 | 83.37 |
| | 10 | 97.46 | 92.71 | 83.23 |
| | 20 | **97.50** | **93.02** | **83.44** |
| $\text{GPT2LM}_r$ | 0 | **97.36** | **92.07** | 79.11 |
| | 10 | 96.93 | 90.80 | 80.76 |
| | 20 | 96.79 | 90.66 | **80.94** |

Table 14: Results of tuning $k_p$ for training in task (b). The best score in each metric among different $k_p$ for training is indicated in **bold**.

| Model | $k_p$ for prediction | Span | Nuclearity | Relation |
|---|---|---|---|---|
| Oracle | 5 | 98.66 | 96.41 | 92.11 |
| | 10 | 99.30 | 98.03 | 94.43 |
| | 20 | **99.47** | **98.48** | **95.42** |
| LMGC$_r$ | 5 | **97.50** | **93.02** | **83.44** |
| | 10 | 97.50 | 92.46 | 83.30 |
| | 20 | 97.29 | 92.25 | 83.30 |
| GPT2LM$_r$ | 5 | **96.79** | **90.66** | **80.94** |
| | 10 | 94.26 | 81.08 | 70.82 |
| | 20 | 93.27 | 77.20 | 66.67 |

Table 15: Results of tuning $k_p$ for prediction in task (b). The best score in each metric among different $k_p$ for prediction is indicated in **bold**.

| Model | $k_s$ for prediction | Precision | Recall | $F_1$ |
|---|---|---|---|---|
| Oracle | 5 | 99.93 | 99.65 | 99.79 |
| | 10 | 99.93 | 99.65 | 99.79 |
| | 20 | 99.93 | 99.65 | 99.79 |
| LMGC$_e$ | 5 | **97.96** | **97.74** | **97.85** |
| | 10 | 97.32 | 97.39 | 97.36 |
| | 20 | 97.33 | 97.53 | 97.43 |
| GPT2LM$_e$ | 5 | **96.94** | **95.91** | **96.42** |
| | 10 | 96.45 | 95.63 | 96.04 |
| | 20 | 95.75 | 95.35 | 95.55 |

Table 16: Results of tuning $k_s$ for prediction in task (c). The best score in each metric among different $k_s$ for prediciton is indicated in **bold**.

| Model | $k_p$ for prediction | Span | Nuclearity | Relation |
|---|---|---|---|---|
| Oracle | 5 | 95.05 | 92.95 | 89.02 |
| | 10 | 95.93 | 94.73 | 91.25 |
| | 20 | **96.21** | **95.36** | **92.45** |
| LMGC$_r$ | 5 | **94.39** | **90.12** | **80.88** |
| | 10 | 94.39 | 89.45 | 80.74 |
| | 20 | 94.18 | 89.24 | 80.63 |
| GPT2LM$_r$ | 5 | **93.65** | **87.80** | **78.59** |
| | 10 | 91.18 | 78.55 | 68.99 |
| | 20 | 90.30 | 74.96 | 65.19 |

Table 17: Results of tuning $k_p$ for prediction in task (c). The best score in each metric among different $k_p$ for prediciton is indicated in **bold**.