Adapting Unsupervised Syntactic Parsing Methodology for Discourse Dependency Parsing

Liwen Zhang[1,2,3,4], **Ge Wang**[1,2,3,4], **Wenjuan Han**[5], **Kewei Tu**[1*]
[1]School of Information Science and Technology, ShanghaiTech University
[2]Shanghai Engineering Research Center of Intelligent Vision and Imaging
[3]Shanghai Institute of Microsystem and Information Technology
[4]University of Chinese Academy of Sciences
[5]Beijing Institute for General Artificial Intelligence, Beijing, China
{zhanglw1, wangge, tukw}@shanghaitech.edu.cn
hanwenjuan@bigai.ai

Abstract

One of the main bottlenecks in developing discourse dependency parsers is the lack of annotated training data. A potential solution is to utilize abundant unlabeled data by using unsupervised techniques, but there is so far little research in unsupervised discourse dependency parsing. Fortunately, unsupervised syntactic dependency parsing has been studied for decades, which could potentially be adapted for discourse parsing. In this paper, we propose a simple yet effective method to adapt unsupervised syntactic dependency parsing methodology for unsupervised discourse dependency parsing. We apply the method to adapt two state-of-the-art unsupervised syntactic dependency parsing methods. Experimental results demonstrate that our adaptation is effective. Moreover, we extend the adapted methods to the semi-supervised and supervised setting and surprisingly, we find that they outperform previous methods specially designed for supervised discourse parsing. Further analysis shows our adaptations result in superiority not only in parsing accuracy but also in time and space efficiency.

1 Introduction

Discourse parsing, aiming to find how the text spans in a document relate to each other, benefits various down-stream tasks, such as machine translation evaluation (Guzmán et al., 2014; Joty et al., 2014), summarization (Marcu, 2000; Hirao et al., 2013), sentiment analysis (Bhatia et al., 2015; Huber and Carenini, 2020) and automated essay scoring (Miltsakaki and Kukich, 2004; Burstein et al., 2013). Researchers have made impressive progress on discourse parsing from the constituency perspective, which presents discourse structures as constituency trees (Ji and Eisenstein, 2014; Feng and Hirst, 2014; Joty et al., 2015; Nishida and Nakayama, 2020). However, as demonstrated by Morey et al. (2018), discourse structure can also be formulated as a dependency structure. Besides that, there might exist ambiguous parsing in terms of the constituency perspective (Morey et al., 2018). All of these suggest that dependency discourse parsing is a different promising approach for discourse parsing.

One of the main bottlenecks in developing discourse dependency parsing methods is the lack of annotated training data since the labeling effort is labor-intensive and time-consuming, and needs well-trained experts with linguistic knowledge (Marcu et al., 1999). This problem can be tackled by employing unsupervised and semi-supervised methods that can utilize unlabeled data. However, while unsupervised methodology has been studied for decades in syntactic dependency parsing, there is little attention paid to the counterpart in discourse dependency parsing. Considering the similarity between syntactic and discourse dependency parsing, it is natural to suggest such methodology can be adapted from the former to the latter.

In this paper, we propose a simple yet effective adaptation method that can be readily applied to different unsupervised syntactic dependency parsing approaches. Adaptation from syntactic dependency parsing to discourse dependency parsing has two challenges. First, unlike syntactic parsing which has a finite vocabulary, in discourse parsing, the number of elementary discourse units (EDUs) is unlimited. This makes it difficult if not impossible to directly apply syntactic approaches requiring enumeration of words or word categories to discourse parsing. Second, in a discourse dependency parse tree, the dependencies within a sentence or a paragraph often form a complete subtree. There is no correspondence to this constraint in syntactic parsing approaches. To address these two chal-

*Corresponding author.

Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, pages 5782–5794
August 1–6, 2021.

lenges, we cluster the EDUs to produce clusters resembling Part-Of-Speech (POS) tags in syntactic parsing and we introduce the Hierarchical Eisner algorithm that finds the optimal parse tree conforming to the constraint.

We applied our adaptation method to two state-of-the-art unsupervised syntactic dependency parsing models: Neural Conditional Random Field Autoencoder (NCRFAE, Li and Tu (2020)) and Variational Variant of Discriminative Neural Dependency Model with Valences (V-DNDMV, Han et al. (2019)). In our experiments, the adapted models performs better than the baseline on both RST Discourse Treebank (RST-DT, Carlson et al. (2001)) and SciDTB (Yang and Li, 2018) in the unsupervised setting. When we extend the two models to the semi-supervised and supervised setting, we find they can outperform previous methods specially designed for supervised discourse parsing.

Further analysis indicates that the Hierarchical Eisner algorithm shows superiority not only in parsing accuracy but also in time and space efficiency. Its empirical time and space complexity is close to $O(n^2)$ with n being the number of EDUs, while the unconstrained algorithm adopted by most previous work has a complexity of $O(n^3)$. The code and trained models can be found at: https://github.com/Ehaschia/DiscourseDependencyParsing.

2 Related Work

Unsupervised syntactic dependency parsing Unsupervised syntactic dependency parsing is the task to find syntactic dependency relations between words in sentences without guidance from annotations. The most popular approaches to this task are Dependency Model with Valences (DMV, Klein and Manning (2004)), a generative model learning the grammar from POS tags for dependency predictions, and its extensions. Jiang et al. (2016) employ neural networks to capture the similarities between POS tags ignored by vanilla DMV and Han et al. (2019) further amend the former with discriminative information obtained from an additional encoding network. Besides, there are also some discriminative approaches modeling the conditional probability or score of the dependency tree given the sentence, such as the CRF autoencoder method proposed by Cai et al. (2017).

Discourse dependency parsing There is limited work focusing on discourse dependency parsing. Li et al. (2014) proposes an algorithm to convert constituency RST tree to dependency structure. In their algorithm, each non-terminal is assigned with a head EDU, which is the head EDU of its left-most nucleus child. Then, a dependency relation is created for each non-terminal from its head to its dependent, in a procedure similar to those designed for syntactic parsing. Hirao et al. (2013) proposes another method that differs from the previous one in the processing of multinuclear relations. Yoshida et al. (2014) proposes a dependency parser built around a Maximum Spanning Tree decoder and trains on dependency trees converted from RST-DT. Their parser achieved better performance on the summarization task than a similar constituency-based parser. Morey et al. (2018) reviews the RST discourse parsing from the dependency perspective. They adapt the the best discourse constituency parsing models until 2018 to the dependency task. Yang and Li (2018) constructs a discourse dependency treebank SciDTB for scientific abstracts. To the best of our knowledge, we are the first to investigate unsupervised and semi-supervised discourse dependency parsing.

Unsupervised Constituent Discourse Parsing Kobayashi et al. (2019) propose two unsupervised methods that build unlabeled constituent discourse trees by using the CKY dynamic programming algorithm. Their methods build the optimal tree in terms of a similarity (dissimilarity) score function that is defined for merging (splitting) text spans into larger (smaller) ones. Nishida et al. (2020) use Viterbi EM with a margin-based criterion to train a span-based neural unsupervised constituency discourse parser. The performance of these unsupervised methods is close to that of previous supervised parsers.

3 Adaptation

We propose an adaptation method that can be readily integrated with different unsupervised syntactic dependency parsing approaches. First, we cluster the element discourse units (EDU) to produce clusters resembling POS tags or words used in syntactic parsing. This is necessary because many unsupervised syntactic parsers require enumeration of words or word categories, typically in modeling multinomial distributions as we shall see in Section 4. While EDUs, which are sequences of words, cannot be enumerated, its clusters can. During parsing, we apply the Hierarchical Eisner algorithm used for parse tree, a novel modified ver-

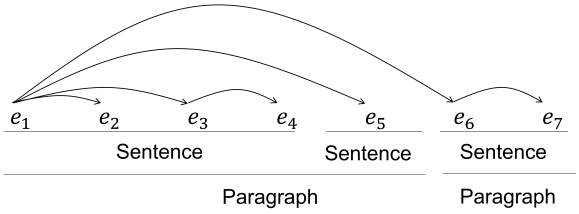


Figure 1: [THE FINANCIAL ACCOUNTING STANDARDS BOARD'S coming rule on disclosure]$_{e_1}$[involving financial instruments]$_{e_2}$[will be effective for financial statements with fiscal years]$_{e_3}$[ending after June 15, 1990.]$_{e_4}$[The date was misstated in Friday's edition .]$_{e_5}$[(See: "FASB Plans Rule on Financial Risk of Instruments"]$_{e_6}$[–WSJ Oct. 27, 1989)]$_{e_7}$

sion of the classic Eisner algorithm, used for parse tree to produce discourse dependency parse trees that conform to the constraint that every sentence or paragraph should correspond to a complete subtree.

3.1 Clustering

Given an input document represented as an EDU sequence $x_1, x_2, \ldots, x_n$, we can use word embedding or context sensitive word embedding to get the vector representation $\mathbf{x}_i$ of the i-th EDU x_i. Specifically, we use BERT (Devlin et al., 2019) to encode each word. Let $\mathbf{w}_i$ be the encoding of the i-th word in the document. For an EDU x_i spanning from word position b to e, we follow Toshniwal et al. (2020) and concatenate the encoding of the endpoints to form its representation: $\mathbf{x}_i = [\mathbf{w}_b; \mathbf{w}_e]$. With the representations of all EDUs from the whole training corpus obtained, we use K-Means (Lloyd, 1982) to cluster them. Let c_i be the cluster label of x_i.

3.2 Hierarchical Eisner Algorithm

The Eisner algorithm (Eisner, 1996) is a dynamic programming algorithm widely used to find the optimal syntactic dependency parse tree. The basic idea of it is to parse the left and right dependents of an token independently and combine them at a later stage. Algorithm 1 shows the pseudo-code of the Eisner algorithm. Here $C_{i\to j}$ represents a *complete span*, which consists of a head token i and all of its descendants on one side, and $I_{i\to j}$ represent an *incomplete span*, which consists of a head i and its partial descendants on one side and can be extended by adding more descendants to that side.

Discourse dependency parse trees, however,

Algorithm 1 Eisner Algorithm

Inputs:
  score matrix $\mathbf{s} \in R^{n\times n}$
Initialize:
  $C = \{\}, I = \{\}$,
  $C_{i\to i} = 0, \quad i = 1, \ldots, n$
for $l = 1, ..., n$ **do** ▷ span length
  for $i = 1, ...n - l$ **do** ▷ span start index
    $j = i + l$ ▷ span end index
    $I_{i\to j} = \max_{i\le k\le j}(s_{ij} + C_{i\to k} + C_{k+1\leftarrow j})$
    $I_{i\leftarrow j} = \max_{i\le k\le j}(s_{ji} + C_{i\to k} + C_{k+1\leftarrow j})$
    $C_{i\to j} = \max_{i\le k\le j}(I_{i\to k} + C_{k\to j})$
    $C_{i\leftarrow j} = \max_{i\le k\le j}(C_{k\to i} + I_{j\to k})$
  end for
end for

Ratio	Train	Dev.	Test
RST-DT	2.6	-	3.0
SciDTB	0.12	0.14	0.14

Table 1: The percentage of dependencies violating the constraint that each sentence or paragraph corresponds to a subtree.

demonstrate structural characteristics not taken into account by the Eisner algorithm. Specifically, a document has a hierarchical structure which divides the document into paragraphs, each paragraph into sentences, and finally each sentence into EDUs, and the discourse parse tree should be consistent with this hierarchical structure. Equivalently, in a discourse parse tree, every sentence or paragraph should be exactly covered by a complete subtree, like Figure 1. We empirically find that this constraint is satisfied by most of the gold discourse parses in the RST Discourse Treebank (RST-DT, Carlson et al. (2001)) and SciDTB (Yang and Li, 2018) datasets (Table 1).

We therefore propose the Hierarchical Eisner algorithm, a novel modification to the Eisner algorithm that incorporates the constraint. Our new algorithm has almost the same state transition formulas as the Eisner algorithm except for a few changes brought by the hierarchical constraint. Concretely, our algorithm finds the optimal parse tree in a bottom-up way and divides the process into 3 steps: intra-sentence parsing, intra-paragraph parsing, and intra-document parsing. In the intra-sentence parsing step, we run the original Eisner algorithm, except that we need not to form a tree. Then in the

Algorithm 2 Modification to Algorithm 1

6: $I_{i\to j} = \max_{i\le k\le j}(s_{ij} + C_{i\to k} + C_{k+1\leftarrow j})$

7: $I_{i\leftarrow j} = \max_{i\le k\le j}(s_{ji} + C_{i\to k} + C_{k+1\leftarrow j})$

8: $C_{i\to j} = \max_{\substack{i\le k\le j \\ j\in E}}(I_{i\to k} + C_{k\to j})$ ▷ Here E is a set of the index of the end boundary of sentences.

9: $C_{i\leftarrow j} = \max_{\substack{i\le k\le j \\ i\in B}}(C_{i\leftarrow k} + I_{k\leftarrow j})$ ▷ Here B is a set of the index of the begin boundary of sentences.

intra-paragraph step, we combine all intra-sentence spans in the paragraph. Under the constraint that there can only be one EDU in every sentence whose head is not belong to this sentence. To achieve that, we modify the state transition equations (step 6-9 in Algorithm 1) to prune invalid arcs. Figure 2 shows some cases during merge across sentence spans. Case 1 are valid because the constraint is satisfied. Case 2 is invalid because the head of EDU e_6 can not be e_4 or e_5 hence the constraint is violated. From these cases, we can find that for incomplete span $I_{i\to k}$ and complete span $C_{k\to j}$ across sentences, we only merge them when j is at the end boundary of a sentence as Algorithm 2 shows. After the intra-paragraph step, we move to the intra-document step to combine paragraph-level spans following the same procedure as in the intra-paragraph step and form the final document-level tree.

Our method has lower time complexity than the original Eisner algorithm. Suppose a document has k_p paragraphs, each paragraph has k_s sentences and each sentence has k_e EDUs. The time complexity of the original Eisner algorithm is $O(k_p^3 k_s^3 k_e^3)$ while the time complexity of our Hierarchical Eisner algorithm is $O(k_p^2 k_s^3 k_e^3)$.

4 Model

We adapt two current state-of-the-art models in unsupervised syntactic dependency parsing for discourse parsing. One is Neural CRF Autoencoder (NCRFAE, Li and Tu (2020); Cai et al. (2017)), a discriminative model, and the other is : Variational Variant of DNDMV (V-DNDMV, Han et al. (2019)), a generative model.

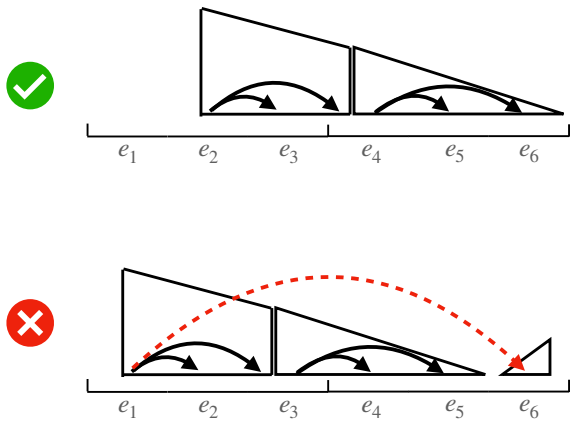


Figure 2: Cases of span merging in discourse parsing. e_1-e_6 are EDUs. red e_1-e_3 make up a sentence and e_4-e_6 make up another sentence. Complete spans are depicted as triangles and incomplete spans as trapezoids.

4.1 Neural CRF Autoencoder

A CRF autoencoder (Ammar et al., 2014) consists of an encoder and a decoder. The encoder predicts a hidden structure, such as a discourse dependency tree in our task, from the input and the decoder tries to reconstruct the input from the hidden structure. In a neuralized CRF autoencoder, we employ neural networks as the encoder and/or decoder.

We use the widely used biaffine dependency parser (Dozat and Manning, 2017) as the encoder to compute the hidden structure distribution $P_\Phi(\mathbf{y}|\mathbf{x})$, parameterized with Φ. Here $\mathbf{y}$ represents the hidden structure and $\mathbf{x}$ is input document. We feed the input document $\mathbf{x}$ into a Bi-LSTM network to produce the contextual representation of each EDU segmentation $\mathbf{r}_i$, and then feed $\mathbf{r}_i$ to two MLP networks to produce two continuous vectors $\mathbf{v}_i^{(head)}$ and $\mathbf{v}_i^{(dep)}$, representing i-th EDU segmentation being used as dependency head and dependent respectively.

A biaffine function is used to compute the score matrix $\mathbf{s}$. Each matrix element s_{ij} , the score for a dependency arc pointing from x_i to x_j, is computed as follows:

$$s_{ij} = \mathbf{v}_i^{(head)\top}\mathbf{W}\mathbf{v}_i^{(dep)} + b \qquad (1)$$

where $\mathbf{W}$ is the parameter matrix and b is the bias.

Following Dozat and Manning (2017) we formulate $P_\Phi(\mathbf{y}|\mathbf{x})$ as a head selection problem process that selects the dependency head of each EDU in-

dependently:

$$P_{\Phi}(\mathbf{y}|\mathbf{x}) = \prod_{i} P(h_i|\mathbf{x}) \tag{2}$$

where h_i is the index of the head of EDU x_i and $P(h_i|\mathbf{x})$ is computed by softmax function with score s_{ij}:

$$P(h_i = j|\mathbf{x}) = \frac{e^{s_{ji}}}{\sum_{k=1}^{n} e^{s_{ki}}} \tag{3}$$

The decoder parameterized with Λ computes $P_{\Lambda}(\hat{\mathbf{x}}|\mathbf{y})$, the probability of the reconstructed document $\hat{\mathbf{x}}$ given the parse tree $\mathbf{y}$. Following Cai et al. (2017) and Li and Tu (2020), we independently predict each EDU $\hat{x}_i$ from its head specified by $\mathbf{y}$. Since EDUs cannot be enumerated, we reformulate the process as predicting the EDU cluster $\hat{c}_i$ given its dependency head cluster c_{h_i}. Our decoder simply specifies a categorical distribution $P(\hat{c}_i|c_{h_i})$ for each possible EDU cluster and compute the reconstruction probability as follows:

$$P_{\Lambda}(\hat{\mathbf{x}}|\mathbf{y}) = \prod_{i} P(\hat{c}_i|c_{h_i}) \tag{4}$$

We achieve the final reconstruction distribution by cascading the encoder and decoder distribution:

$$P_{\Phi,\Lambda}(\hat{\mathbf{x}}, \mathbf{y}|\mathbf{x}) = P_{\Lambda}(\hat{\mathbf{x}}|\mathbf{y}) P_{\Phi}(\mathbf{y}|\mathbf{x}) \tag{5}$$

The best parsing is obtained by maximizing $P_{\Phi,\Lambda}(\hat{\mathbf{x}}, \mathbf{y}|\mathbf{x})$:

$$\mathbf{y}^* = \arg\max_{\mathbf{y}} P_{\Phi,\Lambda}(\hat{\mathbf{x}}, \mathbf{y}|\mathbf{x}) \tag{6}$$

We consider the general case of training the CRF autoencoder with dataset $\mathcal{D}$ containing both labelled data $\mathbb{L}$ and unlabelled data $\mathbb{U}$. Purely supervised or unsupervised learning can be seen as special cases of this setting. The loss function $\mathcal{L}(\mathcal{D})$ consists of a labelled loss $\mathcal{L}_l(\mathbb{L})$ and an unlabelled loss $\mathcal{L}_u(\mathbb{U})$:

$$\mathcal{L}(\mathcal{D}) = \alpha \mathcal{L}_l(\mathbb{L}) + (1 - \alpha)\mathcal{L}_u(\mathbb{U}) \tag{7}$$

where α is the hyperparameter weighting the importance of the two parts.

For the labelled data, where the gold parse trees y^* are known, labelled loss is:

$$\mathcal{L}_l(\mathbb{L}) = -\sum_{\mathbf{x} \in \mathbb{L}} \log P_{\Phi,\Lambda}(\hat{\mathbf{x}}, \mathbf{y}^*|\mathbf{x}) \tag{8}$$

For the unlabelled data where the gold parses are unknown, the unlabelled loss is:

$$\mathcal{L}_u(\mathbb{U}) = -\sum_{\mathbf{x} \in \mathbb{U}} \max_{\mathbf{y} \in \mathcal{Y}(\mathbf{x})} \log P_{\Phi,\Lambda}(\hat{\mathbf{x}}, \mathbf{y}|\mathbf{x}) \tag{9}$$

We optimize the encoder parameter $\mathbf{\Phi}$ and decoder parameter $\mathbf{\Lambda}$ together with gradient descent methods.

4.2 Variational Variant of DNDMV

V-DNDMV is a variational autoencoder model composed of both an encoder and a decoder. The encoder is a Bi-LSTM that takes the input document and produces parameters of a Gaussian distribution from which a continuous vector $\mathbf{s}$ summarizing the document sampled.

The decoder models the joint probability of the document and its discourse dependency tree condition on $\mathbf{s}$ with a generative grammar. The grammar is defined on a finite set of discrete symbols, so in our adapted model, input documents are represented by EDU clusters instead of EDUs that are infinite in number. There are three types of grammar rules, each associated with a set of probabilistic distributions: ROOT,CHILD and DECISION. To generate a document, we firstly sample from the ROOT distribution $\mathrm{P}_{\text{ROOT}}(chd|s)$ to determine the cluster label of the head EDU of the document and then recursively decide whether to generate a new child EDU cluster and what child EDU cluster to generate by sampling from the DECISION distribution $\mathrm{P}_{\text{DECISION}}(dec|h, dir, val, s)$ and CHILD distribution $\mathrm{P}_{\text{CHILD}}(chd|h, dir, val, s)$. dir denotes the generation direction (i.e, left or right), val is a binary variable denoting whether the current EDU already has a child in the direction dir or not. dec is a binary variable indicating whether to continue generating a child EDU, and h and chd denote the parent and child EDU cluster respectively. We use neural networks to calculate these distributions. The input of the networks is the continuous vector or matrix representations of grammar rule components such as h, chd, val and dir as well as document vector $\mathbf{s}$ produced by the encoder.

The training objective for learning the model is the probability of the training data. The intermediate continuous vector $\mathbf{s}$ and the hidden variable representing the dependency tree are both marginalized. Since the marginalized probability cannot be calculated exactly, V-DNDMV maximizes the Evidence Lower Bound (ELBO), a lower bound of the marginalized probability. ELBO consists of

the conditional likelihood of the training data and an regularisation term given by the KL divergence between $P_\Theta(\mathbf{s}|\mathbf{x})$ and $P(\mathbf{s})$ (which is a standard Gaussian). The conditional likelihood is shown as follows:

$$\mathcal{L}(\Theta) = \frac{1}{N}\sum_{i=1}^{N} \sum_{\mathbf{y}^{(i)} \in \mathcal{Y}(\mathbf{x}^{(i)})} \log P_\Theta(\mathbf{x}^{(i)}, \mathbf{y}^{(i)}|\mathbf{s}^{(i)}) \tag{10}$$

Here N is the number of training samples, $\mathbf{y}$ is the dependency tree and $\mathcal{Y}(\mathbf{x})$ is the set of all possible dependency tree in $\mathbf{x}$. Θ is the parameters of the neural networks. We can rewrite the conditional probability as following:

$$P_\Theta(\mathbf{x}, \mathbf{y}|\mathbf{s}) = \prod_{\mathbf{r} \in (\mathbf{x}, \mathbf{y})} P(\mathbf{r}|\mathbf{s}) \tag{11}$$

where $\mathbf{r}$ is the grammar rule involved in generating $\mathbf{x}$ along with $\mathbf{y}$.

We optimize ELBO using the expectation-maximization (EM) algorithm, alternating the E-step and the M-step. In the E-step, we fix rule parameters and use our Hierarchical Eisner algorithm to compute the expectation of possible dependency tree $\mathbf{y}$, which gives the expected count of rules used in the training samples. In the M-step, expected count of rules computed in the E-step is used to train the prediction neural networks with gradient descent methods. The regularisation term is also optimized using gradient descent methods in the M-step. After training, the parsing result $\mathbf{y}^*$of a new test case $\mathbf{x}$ is obtained as:

$$\mathbf{y}^* = \arg\max_{\mathbf{y} \in \mathcal{Y}(\mathbf{x})} P_\Theta(\mathbf{x}, \mathbf{y}|\mathbf{s}) \tag{12}$$

5 Experiment

5.1 Setting

Data We evaluate the performance of our models on the RST Discourse Treebank* (RST-DT, Carlson et al. (2001)) and SciDTB† (Yang and Li, 2018). RST-DT consists of Wall Street Journal articles manually annotated with RST structures (Mann and Thompson, 1988). We use the method proposed by Li et al. (2014) to convert the RST structure samples into dependency structures. SciDTB consists of scientific abstracts from ACL Anthology annotated with dependency structures.

*https://catalog.ldc.upenn.edu/LDC2002T07

†https://github.com/PKU-TANGENT/SciDTB

Hyper-parameter For our NCRFAE model, we adopt the hyper-parameters of Li and Tu (2020). For our V-NDNMV model we adopt the hyper-parameters of Han et al. (2019). We use Adam (Kingma and Ba, 2015) to optimize our objective functions. Experimental details are provided in Appendix A.

5.2 Main Result

We compared our methods with the following baselines:

Right Branching (RB) is a rule based method. Given a sequence of elements (i.e., EDUs or subtrees), RB generates a left to right chain structure, like $x_1 \rightarrow x_2$, $x_2 \rightarrow x_3 \cdots$. In order to develop a strong baseline, we include the hierarchical constraint introduced in Section 3.2 in this procedure. That is, we first build sentence-level discourse trees using the right branching method based on sentence segmentation. Then we build paragraph-level trees using the right branching method to form a left to right chain of sentence-level subtrees. Finally we obtain document-level trees in the same way. Since this method has three stages, we call it "**RB_RB_RB**". This simple procedure forms a strong baseline in terms of performance. As Nishida and Nakayama (2020) reports, the unlabeled F1 score of constituent structures of RB_RB_RB reaches 79.9 on RST-DT. Correspondingly, the performance of the supervised method proposed by (Joty et al., 2015) is 82.5.

NISHIDA20 is a neural model for unsupervised discourse constituency parsing proposed by Nishida and Nakayama (2020). This model runs a CKY parser that uses a Bi-LSTM model to learn representations of text spans, complemented with lexical, syntactic and structural features. We convert its result to dependency structure using the same conversation method of Li et al. (2014). To make a fair comparison, we use RB_RB_RB to initialize their model instead of RB*_RB_RB as in their paper, where RB* means using predicted syntactic structures for initialization at the sentence level.

Compared with baselines , our two adapted models NCRFAE and V-DNDMV both achieve better performance on the two datasets. Results also show that the generative model V-DNDMV is better than the discriminatve model NCRFAE in the unsupervised setting.

We also investigate the semi-supervised setting

	SciDTB	RST-DT
RB_RB_RB	52.5	43.9
NISHIDA20	-	41.9
Adapted V-DNDMV	**54.4**	**44.2**
Adapted NCRFAE	53.3	44.0

Table 2: Unsupervised discourse dependency parsing results on RST-DT and SciDTB. The evaluation metric is the Unlabeled Attachment Score (UAS).

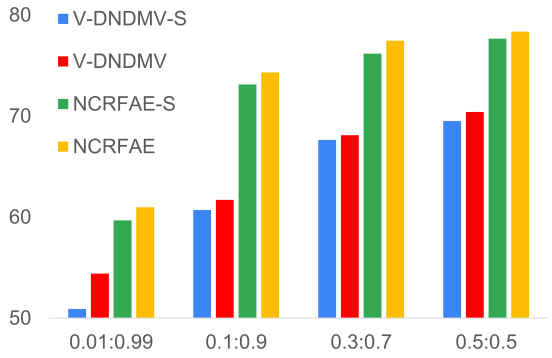


Figure 3: Semi-supervised discourse dependency parsing results on SciDTB. The V-DNDMV-S and NCRFAE-S mean these two model are trained on labeled data only. The x-axis represents the ratio of labeled/unlabeled data used for training. The y-axis represents the UAS score.

on the SciDTB dataset of our adapted models with varied ratios of labeled/unlabeled data. Experimental results are shown in Figure 3, which indicate that NCRFAE outperforms V-DNDMV for all the ratios. Even when trained with only a few labeled data (0.01 of labeled data in SciDTB, only about 7 samples), the discriminative model already outperforms the generative model significantly. Besides that, we also find our semi-supervised methods reach higher UAS scores than their supervised versions (trained with labeled data only) for all the labeled/unlabeled data ratios.

Inspired by the promising results in the semi-supervised setting, we also investigate the performance of our adapted NCRFAE and V-DNDMV in the fully supervised setting. The results are shown in Table 3. We evaluate our models on the RST-DT and SciDTB datasets and compare them with eight models. NIVRE04 (Nivre et al., 2004) and WANG17 (Wang et al., 2017) are two transition-based models for dependency parsing. Yang and Li (2018) adapts them to discourse dependency parsing. FENG14 (Feng and Hirst, 2014), JI14

[‡]We correct their evaluation metrics, so the result is different from the original paper (Li et al., 2014).

	RST-DT		SciDTB	
	UAS	LAS	UAS	LAS
NIVRE04	-	-	70.2	53.5
LI14	48.7[‡]	-	57.6	42.5
FENG14	65.6	48.5	-	-
JI14	66.9	51.7	-	-
JOTY15	64.4	48.0	-	-
BRAUD17	66.1	49.9	-	-
WANG17	-	-	70.2	54.5
MOREY18	66.4	48.7	-	-
Adapted V-DNDMV	63.5	-	73.4	-
Adapted NCRFAE	**70.2**	**51.8**	**79.1**	**65.0**

Table 3: Supervised discourse dependency parsing results on RST-DT and SciDTB. The UAS is Unlabeled Attachment Score and LAS is Labeled Attachment Score.

(Ji and Eisenstein, 2014), JOTY15 (Joty et al., 2015) and BRAUD17 (Braud et al., 2017) are methods for discourse constituent parsing and they are adapted for discourse dependency parsing by Morey et al. (2018). LI14 (Li et al., 2014) and MOREY18 (Morey et al., 2018) are graph-based and transition-based methods specially designed for discourse dependency parsing, respectively. These models are statistical or simple neural models, and they do not use pretrained language models (like BERT, ELMo (Peters et al., 2018)) to extract features.

As Table 3 shows, the performance of our NCRFAE is significantly better than the baseline models. Especially, the UAS and LAS of NCRFAE are 8.9 points and 11.5 points higher than the best baseline models on the SciDTB dataset, respectively. Besides that, we find that V-DNDMV also beats baselines on the SciDTB dataset and reaches comparable results on RST-DT. We also test our approaches without using BERT and find that they still outperform the baselines. For example, the performance of NCRFAE with GloVe (Pennington et al., 2014) on Scidtb averaged over 5 runs is: UAS: 73.9 LAS: 55.5. These results again give evidence for our success in adapting unsupervised syntactic dependency parsing methods for discourse dependency parsing as the adapted methods not only work in the unsupervised setting, but also reach state-of-the-art in the supervised setting.

As for the performance gap between V-DNDMV and NCRFAE, we believe that the main reason is their different abilities to extract contextual features from the input text for the parsing task. As a generative model, the decoder of V-DNDMV follows

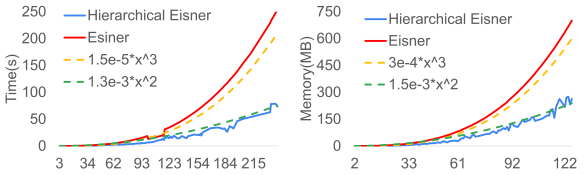


Figure 4: Analysis of time and space cost in running our hierarchical Eisner and traditional Eisner algorithm on RST-DT dataset against document length.Left: time cost. Right: space cost.

a strong assumption that each token in the input text is generated independently, which prevents the contextual features from being directly used. Instead, contextual features are mixed with other information in the document representation which acts as the condition of the generation process in the model. NCRFAE, on the other hand, employs a discriminative parser to leverage contextual features for dependency structure prediction directly. Thus, as long as there is sufficient labeled data, NCRFAE can achieve much better results than V-DNDMV. We have observed a similar phenomenon in syntactic parsing.

Significance test We investigate the significance of the performance improvement in every setting. For unsupervised parsing, we perform a t-test between the strongest baseline RB_RB_RB and V-DNDMV. The t-value and p-value calculated on 10 runs are 2.86 and 0.00104, which shows the significance of the improvement. For the semi-supervised results, we also perform significance tests between the semi-supervised and supervised-only results. The results show that our semi-supervised method significantly outperforms the supervised-only method. For example, on the 0.5:0.5 setting, the t-value is 2.13 and the p-value is 0.04767. For the fully supervised setting, due to a lack of code from previous work, it is currently difficult for us to carry out a significance analysis. Instead, we show that our models are very stable and consistently outperform the baselines by running our models for 10-times. For example, our NCRFAE UAS score is 78.95±0.29 on the Scidtb dataset.

6 Analysis

6.1 Eisner vs. Hierarchical Eisner

In the left part of Figure 4 we show the curves of the time cost of the hierarchical and traditional Eisner algorithms against the RST-DT document length.

Clusters	10	30	50	100
UAS	52.7	53.9	**54.6**	53.5

Table 4: UAS with different cluster numbers on the development set of Scidtb.

	Mutual Information
Random	0.007[§]
K-means	0.106
NICE	0.096

Table 5: Mutual information

The experiments are run on servers equipped with NVIDIA Titan V GPUs. We can observe clearly that the curve of the Hierarchical Eisner algorithm always stays far below that of the Eisner algorithm, which verifies our theoretical analysis on the time complexity of the hierarchical Eisner algorithm in section 3.2.

The right part of Figure 4 demonstrates a similar phenomenon where we illustrate the memory usage of the hierarchical and traditional Eisner algorithms against the training document length in the same computing environment. From the curves of these two figures we can conclude that our Hierarchical Eisner algorithm has advantage over the traditional one in both time and space efficiencies.

Besides the superiority in computational efficiency, our experiments also indicate that our Hierarchical Eisner algorithm can achieve better performance than the traditional one. With other conditions fixed, the UAS produced by Hierarchical Eisner is 79.1 in the task of supervised discourse parsing on the SciDTB dataset while the corresponding result of the Eisner algorithm is 78.6.

6.2 Number of clusters

To explore the suitable number of clusters of EDUs, we evaluate our NCRFAE model with different cluster numbers from 10 to 100. As table 4 shows, there is an upward trend while the number of clusters increases from 10 to 50. After reaching the peak, the UAS decreases as the number of cluster continues to increase. We thus choose 50 for our experiments.

6.3 Label analysis

In order to inspect if there exist any coherent relations between the clusters of EDUs obtained for

[§]This is the actual evaluation result and the theoretical result should be 0.0

Figure 5: Heat-maps of probabilities that relations use different label as dependency head (left) or child (right).

adaptation in discourse parsing and the labels of dependency arcs, similar to that between POS tags and syntactic dependency labels, we compute the co-appearance distribution of cluster labels and dependency arc labels. In Figure 5, we show the probabilities of the clusters being used as heads $p_{head}(c_k|r_m)$ and children $p_{child}(c_k|r_m)$ given different dependency types respectively. Here c_k and r_m represent different type of clusters and relations. We cluster EDUs to 10 clusters and only show a subset of them. Detailed heat-map can be found in Appendix B.

By observing the two heat-maps, we notice obvious trends that for each dependency arc label, the co-appearance probabilities are concentrated at certain cluster labels. For example, when the cluster is used as dependency heads, more than 60% of the co-appearance probability for arc label COMPARISON and SAME-UNIT is concentrated at cluster type 9 and 6 respectively; when the cluster is used as dependency children, cluster type 1 receives more than 40% of the co-appearance probability for certain arc labels. The property displayed by the adaptation clusters is very similar to that of POS tags, which justifies our clustering strategy adopted for discourse parsing.

To further quantify the coherence between the adaptation clusters and dependency arcs, we evaluate the mutual information between two discrete random variables in the training set of SciDTB: one is the tuple consists of two cluster labels for a pair of EDUs in the training sample, representing dependency head and child respectively; and the other is the binary random variable indicating whether there exists a dependency arc between a EDU pair in the training data. Besides our adaptation clusters, we also evaluate this metric for two other clustering strategies, random clustering and NICE proposed by He et al. (2018), for comparison and show the results in Table 5. We see that measured by mutual information, clusters produced by our clustering strategy is much more coherent with dependencies than the other strategies.

7 Conclusion

In this paper, we propose a method to adapt unsupervised syntactic parsing methods for discourse dependency parsing. First, we cluster the element discourse units (EDU) to produce clusters resembling POS tags. Second, we modify the Eisner algorithm used for finding the optimal parse tree with hierarchical constraint. We apply the adaptations to two unsupervised syntactic dependency parsing methods. Experimental results show that our method successfully adapts the two models for discourse dependency parsing, which demonstrate advantages in both parsing accuracy and running efficiency.

Acknowledgment

This work was supported by the National Natural Science Foundation of China (61976139).

References

Waleed Ammar, Chris Dyer, and Noah A Smith. 2014. Conditional random field autoencoders for unsupervised structured prediction. In *Advances in Neural Information Processing Systems*, volume 27, pages 3311–3319. Curran Associates, Inc.

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

Chloé Braud, Maximin Coavoux, and Anders Søgaard. 2017. Cross-lingual RST discourse parsing. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 292–304, Valencia, Spain. Association for Computational Linguistics.

Jill Burstein, Joel Tetreault, and Martin Chodorow. 2013. Holistic discourse coherence annotation for noisy essay writing. *Dialogue & Discourse*, 4(2):34–52.

Jiong Cai, Yong Jiang, and Kewei Tu. 2017. CRF autoencoder for unsupervised dependency parsing. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1638–1643, Copenhagen, Denmark. Association for Computational Linguistics.

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*.

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.

Timothy Dozat and Christopher D. Manning. 2017. Deep biaffine attention for neural dependency parsing. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Jason M. Eisner. 1996. Three new probabilistic models for dependency parsing: An exploration. In *COLING 1996 Volume 1: The 16th International Conference on Computational Linguistics*.

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

Wenjuan Han, Yong Jiang, and Kewei Tu. 2019. Enhancing unsupervised generative dependency parser with contextual information. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5315–5325, Florence, Italy. Association for Computational Linguistics.

Junxian He, Graham Neubig, and Taylor Berg-Kirkpatrick. 2018. Unsupervised learning of syntactic structure with invertible neural projections. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1292–1302, Brussels, Belgium. Association for Computational Linguistics.

Tsutomu Hirao, Yasuhisa Yoshida, Masaaki Nishino, Norihito Yasuda, and Masaaki Nagata. 2013. Single-document summarization as a tree knapsack problem. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1515–1520, Seattle, Washington, USA. Association for Computational Linguistics.

Patrick Huber and Giuseppe Carenini. 2020. Unsupervised learning of discourse structures using a tree autoencoder. *arXiv preprint arXiv:2012.09446*.

Yangfeng Ji and Jacob Eisenstein. 2014. Representation learning for text-level discourse parsing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13–24, Baltimore, Maryland. Association for Computational Linguistics.

Yong Jiang, Wenjuan Han, and Kewei Tu. 2016. Unsupervised neural dependency parsing. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 763–771, Austin, Texas. 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. 2014. DiscoTK: Using discourse structure for machine translation evaluation. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 402–408, Baltimore, Maryland, USA. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations,*

ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings.

Dan Klein and Christopher Manning. 2004. Corpus-based induction of syntactic structure: Models of dependency and constituency. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, pages 478–485, Barcelona, Spain.

Naoki Kobayashi, Tsutomu Hirao, Kengo Nakamura, Hidetaka Kamigaito, Manabu Okumura, and Masaaki Nagata. 2019. Split or merge: Which is better for unsupervised RST 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 5797–5802, Hong Kong, China. Association for Computational Linguistics.

Sujian Li, Liang Wang, Ziqiang Cao, and Wenjie Li. 2014. Text-level discourse dependency parsing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 25–35, Baltimore, Maryland. Association for Computational Linguistics.

Zhao Li and Kewei Tu. 2020. Unsupervised cross-lingual adaptation of dependency parsers using CRF autoencoders. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2127–2133, Online. Association for Computational Linguistics.

Stuart Lloyd. 1982. Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2):129–137.

William C Mann and Sandra A Thompson. 1988. Rhetorical structure theory: Toward a functional theory of text organization. *Text*, 8(3):243–281.

Daniel Marcu. 2000. *The theory and practice of discourse parsing and summarization*. MIT press.

Daniel Marcu, Estibaliz Amorrortu, and Magdalena Romera. 1999. Experiments in constructing a corpus of discourse trees. In *Towards Standards and Tools for Discourse Tagging*.

Eleni Miltsakaki and Karen Kukich. 2004. Evaluation of text coherence for electronic essay scoring systems. *Natural Language Engineering*, 10(1):25.

Mathieu Morey, Philippe Muller, and Nicholas Asher. 2018. A dependency perspective on RST discourse parsing and evaluation. *Computational Linguistics*, 44(2):197–235.

Kosuke Nishida, Kyosuke Nishida, Itsumi Saito, Hisako Asano, and Junji Tomita. 2020. Unsupervised domain adaptation of language models for reading comprehension. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 5392–5399, Marseille, France. European Language Resources Association.

Noriki Nishida and Hideki Nakayama. 2020. Unsupervised discourse constituency parsing using Viterbi EM. *Transactions of the Association for Computational Linguistics*, 8:215–230.

Joakim Nivre, Johan Hall, and Jens Nilsson. 2004. Memory-based dependency parsing. In *Proceedings of the Eighth Conference on Computational Natural Language Learning (CoNLL-2004) at HLT-NAACL 2004*, pages 49–56, Boston, Massachusetts, USA. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. 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.

Shubham Toshniwal, Haoyue Shi, Bowen Shi, Lingyu Gao, Karen Livescu, and Kevin Gimpel. 2020. A cross-task analysis of text span representations. In *Proceedings of the 5th Workshop on Representation Learning for NLP*, pages 166–176, Online. 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.

An Yang and Sujian Li. 2018. SciDTB: Discourse dependency TreeBank for scientific abstracts. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 444–449, Melbourne, Australia. Association for Computational Linguistics.

Yasuhisa Yoshida, Jun Suzuki, Tsutomu Hirao, and Masaaki Nagata. 2014. Dependency-based discourse parser for single-document summarization. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1834–1839, Doha, Qatar. Association for Computational Linguistics.

A Experimental Details for Our NCRFAE and V-DNDMV

We implement our NCRFAE and V-DNDMV models by Pytorch 1.6 and Python 3.8.3. We run our experiments on a server with Intel(R) Xeon(R) Gold 5115 CPU and NVIDIA Titan V GPU. Based on these software and hardware environments, our NCRFAE and V-DNDMV models trained on the SciDTB dataset use about 30 and 45 minutes, respectively. Moreover, our NCRFAE and V-DNDMV models trained on the RST-DT dataset use about 4 and 18 hours, respectively. The number of parameters in NCRFAE is about 8.26 million, and the number of parameters in V-DNDMV is 0.47 million. The hyperparameter configurations of the result report in our paper are shown in table 6. We choose the hyperparameter configurations by manual tuning and the UAS score on the development dataset is used to select among them. Due to the lack of development set of RST-DT, we prepare a development set with 20 instances randomly sampled from the training set. The size of each dataset is shown in Table 7.

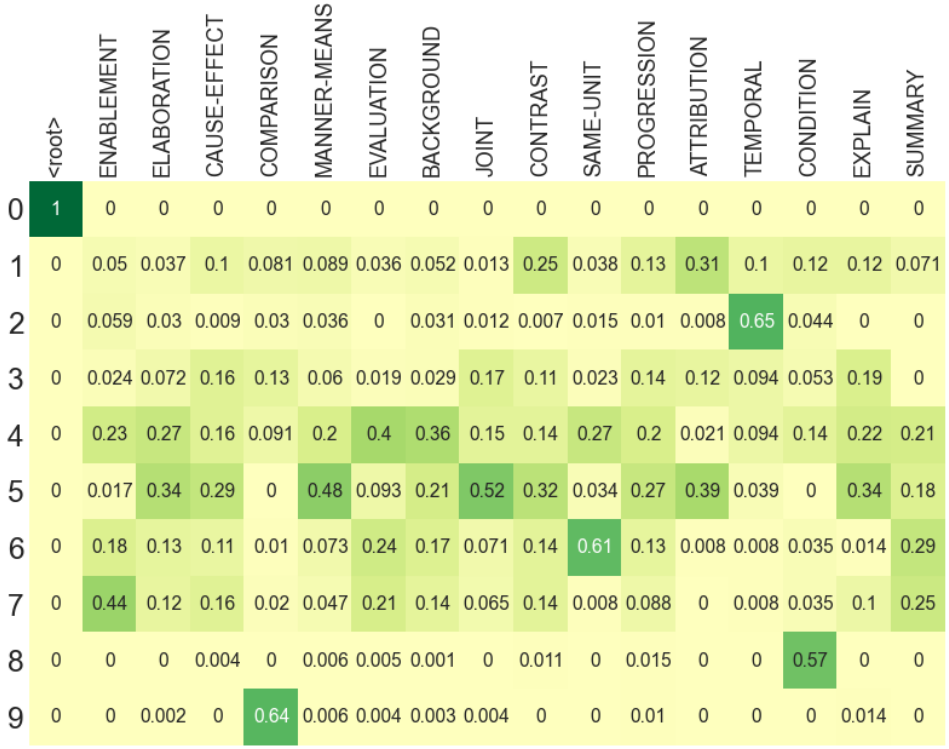
	NCRFAE	V-DNDMV
Cluster		
Cluster Number	50	50
Hidden Layer		
EDU Embedding	1536	1536
Cluster Embedding	-	20
Valence Embedding	-	20
FNN(embedding)	1*200	1*200
Bi-LSTM	1*400	1*32
LSTM dropout	0.33	0.0
FNN(head)	1*500	-
FNN(dep)	1*200	-
FNN dropout	0.33	0.3
Optimizer & Loss		
Learning Rate	2e-3	1e-3
Adam beta 1	0.9	0.9
Adam beta 2	0.9	0.999
l2reg	1e-4	0.0

Table 6: Hyper-parameters for our NCRFAE and V-DNDMV.

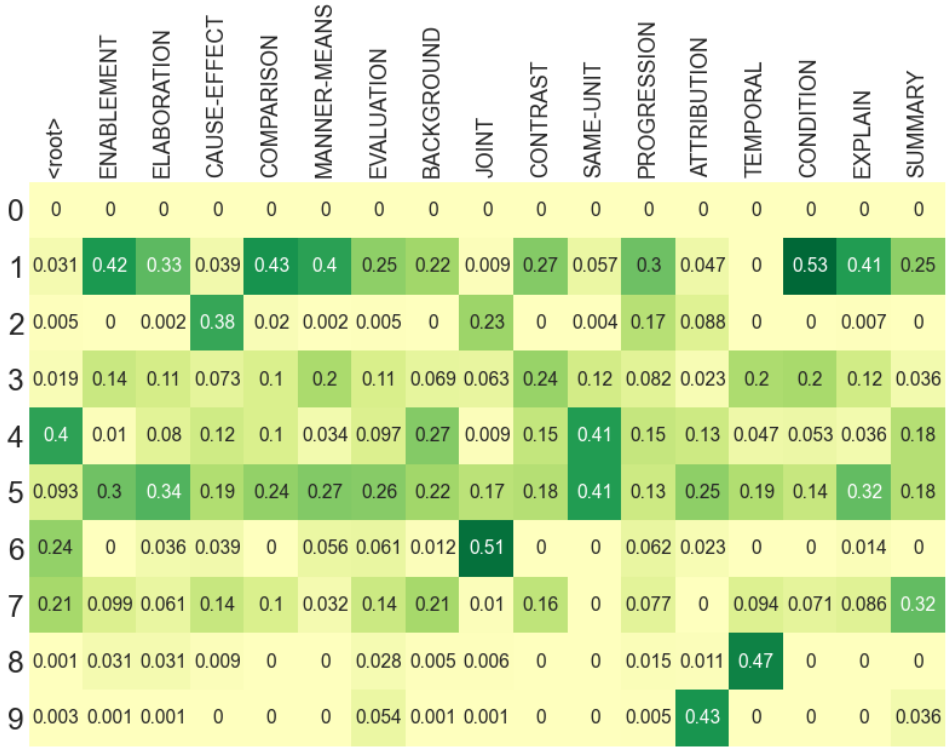
	Usage	Doc.	EDU	Relation Type
RST-DT	Train	347	19443	19
	Test	38	2346	
SciDTB	Train	742	10467	17
	Dev.	152	2018	
	Test	151	2013	

Table 7: Size of RST-DT and SciDTB. Here the relation type is coarse-grained relation.

B Full Heat-maps



(a) Head



(b) Child

Figure 8: Heat-maps of probabilities that relations use different label as dependency head (a) or child (b).