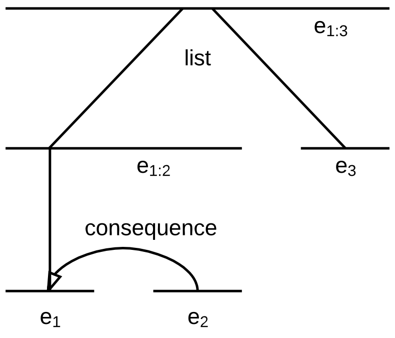
Learning Contextually Informed Representations for Linear-Time Discourse Parsing

Yang Liu and **Mirella Lapata**
Institute for Language, Cognition and Computation
School of Informatics, University of Edinburgh
10 Crichton Street, Edinburgh EH8 9AB
Yang.Liu2@ed.ac.uk,mlap@inf.ed.ac.uk

Abstract

Recent advances in RST discourse parsing have focused on two modeling paradigms: (a) high order parsers which jointly predict the tree structure of the discourse and the relations it encodes; or (b) linear-time parsers which are efficient but mostly based on local features. In this work, we propose a linear-time parser with a novel way of representing discourse constituents based on neural networks which takes into account global contextual information and is able to capture long-distance dependencies. Experimental results show that our parser obtains state-of-the art performance on benchmark datasets, while being efficient (with time complexity linear in the number of sentences in the document) and requiring minimal feature engineering.



[Only a few months ago, the 124-year-old securities firm seemed to be on the verge of a meltdown,]*e₁* **[racked by internal squabbles and defections.]***e₂* **[Its relationship with parent General Electric Co. had been frayed since a big Kidder insider-trading scandal two years ago.]***e₃*

Figure 1: Example text (bottom) composed of two sentences (three EDUs) and its RST discourse tree representation (top).

1 Introduction

The computational treatment of discourse phenomena has recently attracted much attention, due to their increasing importance for potential applications. Knowing how text units can be composed into a coherent document and how they relate to each other e.g., whether they express contrast, cause, or elaboration, can usefully aid downstream tasks such summarization (Yoshida et al., 2014), question answering (Chai and Jin, 2004), and sentiment analysis (Somasundaran, 2010).

Rhetorical Structure Theory (RST, Mann and Thompson 1988), one of the most influential frameworks in discourse processing, represents texts by trees whose leaves correspond to Elementary Discourse Units (EDUs) and whose nodes specify how these and larger units (e.g., multi-sentence segments) are linked to each other by rhetorical relations. Discourse units are further characterized in terms of their importance in text: nuclei denote central segments, whereas satellites denote peripheral ones. Figure 1 shows an example of a discourse tree representing two sentences with three EDUs (e_1,e_2, and e_3). EDUs e_1 and e_2 are connected with a mononuclear relation (i.e., *Consequence*), where e_1 is the nucleus and e_2 the satellite (indicated by the left pointing arrow in the figure). Span $e_{1:2}$ is related to e_3 via *List*, a multi-nuclear relation, expressing the fact that both spans are equally important and therefore both nucleus.

Given such tree-based representations of discourse structure, it is not surprising that RST-style document analysis is often viewed as a parsing task. State-of-the-art performance on RST parsing is achieved by cubic-time parsers (Li, Li, and Hovy, 2014; Li, Li, and Chang, 2016), with $O(n^3)$ time complexity (where n denotes the number of sentences in the document). These systems model the relations between all possible adjacent discourse segments and use a CKY-style algorithm to generate a global optimal tree. The high order complexity renders such parsers inefficient in practice, especially when processing large documents. As a result, more efficient linear-time discourse parsers have been proposed (Feng and Hirst, 2014; Ji and Eisenstein, 2014) which make local decisions and model the structure of the discourse and its relations separately. In this case, features are extracted from a local context (i.e., a small window of discourse constituents) without considering document-level information, which has been previously found useful in discourse analysis (Feng and Hirst, 2012).

In this paper, we propose a simple and efficient linear-time discourse parser with a novel way of learning contextual representations for discourse constituents. To guarantee linear-time complexity, we use a two-stage approach: we first parse each sentence in a document into a tree whose leaves correspond to EDUs, and then parse the document into a tree whose leaves correspond to already preprocessed sentences. The feature learning process for both stages is based on neural network models. At the sentence level, Long-Short Term Memory Networks (LSTMs; Hochreiter and Schmidhuber 1997) learn representations for EDUs and larger constituents, whereas at the document level, LSTMs learn representations for entire sentences. Treating a sentence as a sequence of EDUs and a document as a sequence of sentences allows to incorporate important contextual information on both levels capturing long-distance dependencies.

Recurrent neural networks excel at modeling sequences, but cannot capture hierarchical structure which is important when analyzing multi-sentential discourse. We therefore adopt a more structure-aware representation at the document level which we argue is complementary to the flat representations obtained from the LSTM. We represent documents as trees using recursive neural networks (Socher et al., 2012). Experimental evaluation on the RST Treebank shows that our parser yields comparable performance to previous linear-time systems, without requiring extensive manual feature engineering and improves upon related neural models (Li et al., 2014, 2016) on discourse relation classification, while being more efficient.

The rest of this paper is organized as follows. We overview related work in the following section. We describe the general flow of our parser in Section 3 and provide details on our parsing algorithm and feature learning method in Section 4. Experimental results are reported in Section 5. Section 7 concludes the paper.

2 Related Work

Recent advances in discourse modeling have greatly benefited from the availability of resources annotated with discourse-level information such as the RST Discourse Treebank (RST-DT; Carlson et al. 2003) and the Penn Discourse Treebank (PDTB, Prasad et al. 2008). In this work, we focus on RST-style discourse parsing, where a tree representation is derived for an entire document. In PDTB, discourse relations are annotated mostly between adjacent sentences and no global tree structure is provided.

Early approaches to discourse parsing (Marcu, 2000; LeThanh et al., 2004) have primarily focused on overt discourse markers (or cue words) and used a series of rules to derive the discourse tree structure. Soricut and Marcu (2003) employed a standard bottom-up chart parsing algorithm with syntactic and lexical features to conduct sentence-level parsing. Baldridge and Lascarides (2005) and Sagae (2009) used probabilistic head-driven parsing techniques. Subba and Di Eugenio (2009) were the first to incorporate rich compositional semantics into sentence- and document-level discourse parsing.

HILDA (Hernault et al., 2010) has been one of the most influential document-level discourse parsers paving the way for many machine learning-based models. HILDA parses a document pre-segmented into EDUs with two support vector machine classifiers working iteratively in a pipeline. At each iteration, a binary SVM predicts which adjacent units should be merged and then a multi-class SVM predicts their discourse relation. Subsequent work (Feng and Hirst, 2014; Joty et al., 2013) has shown that two-stage systems are not only efficient but can also achieve competitive performance. CKY-based parsers which guarantee globally optimal results have also been developed (Joty et al., 2013; Li et al., 2014).

Ji and Eisenstein (2014) were the first to apply neural network models to RST discourse parsing; their shift-reduce parser uses a feedforward neural network to learn the representations of the transition stack and queue. Li et al. (2014) proposed a CKY-based parser which uses recursive neural networks to learn representations for EDUs and their composition during the tree-building process. More recently, Li et al. (2016) designed a CKY-based parser which uses LSTMs to model text spans and a tensor-based transformation to compose adjacent spans.

Our own work joins others (Feng and Hirst, 2014; Joty et al., 2013) in adopting a two-stage architecture for our discourse parser. However, rather than considering each discourse constituent independently, we learn contextually-aware representations capturing long-range dependencies across sentences and documents. Discourse constituents at all levels are modeled with recurrent neural networks adopting a relatively simple, yet efficient, architecture compared to previously proposed neural systems (Li et al., 2016). Finally, we experimentally assess whether sequence-based representations are expressive enough by comparing them to those obtained (with recursive neural networks) from structured inputs.

3 Discourse Parser Overview

In RST discourse parsing, a document is first segmented into EDUs and a parser then builds a discourse tree with the EDUs as leaves. The first subtask is considered relatively easy with state-of-art accuracy at above 90% (Hernault et al., 2010). As a result, recent research focuses on the second subtask and often uses manual EDU segmentation.

Joty et al. (2013) found that a two-stage parsing strategy, which separates intra-sentential from multi-sentential parsing, has some advantages for document-level discourse parsing, since the distribution of discourse relations and useful features are different in the two stages. Based on their findings, our model also follows a two-stage approach and is composed by two components: an *intra-sentential parser* and a *multi-sentential parser*. Given a document pre-segmented into EDUs, our intra-sentential parser first builds a sentence-level discourse tree for individual sentences. Then, our multi-sentential parser creates a discourse tree for the entire document.

To guarantee linear-time complexity, both parsers adopt a greedy bottom-up tree-building process (Hernault et al., 2010) and are based on two conditional random field (CRF) models, one for creating discourse structure and another one for assigning relations. The intra-sentential parser considers adjacent EDUs and decides whether they should be connected (based on the scores predicted by the first CRF) and their relation (based on predictions of the second CRF). The multi-sentential parser follows the same procedure while operating over sentences.

The CRFs employ feature representations which we obtain using neural networks. Specifically, discourse constituents at all levels are modeled with recurrent neural networks (see Section 4.2 for details). In addition, for inter-sentential constituents, we complement the flat text span representation with recursive neural networks (see Section 4.4). We argue that the combination is advantageous; a sequential text span representation is in principle unsuitable for capturing hierarchical discourse structure, whereas a tree-based representation can be more precise, albeit less robust (due to the accumulation of errors from the recursive tree-building process).

4 Parsing Model

4.1 Intra-sentential Parser

To parse a sentence, we start with EDUs, which can be viewed as discourse constituents at the first level. As mentioned earlier, our intra-sentential parser is based on two linear-chain CRFs, the structure CRF decides which pair of constituents should be merged at the current level and the relation CRF assigns discourse relations to non-leaf constituents. For example, let $C_1 = \{e_1, e_2, \cdots, e_m\}$ denote a sentence with a sequence of EDUs, where e_i is the i^{th} EDU in the sentence. Suppose the structure CRF decides to merge together e_2 and e_3, then the next-level sequence is $C_2 = \{e_1, e_{2:3}, e_4, \cdots, e_m\}$ and the relation CRF will assign a discourse relation to $e_{2:3}$, the only non-leaf constituent so far. This process iterates until all EDUs are merged and a discourse subtree is generated for the entire sentence.

A linear-chain CRF for intra-sentential discourse parsing is shown in Figure 2. Here, $C = \{c_0, \cdots, c_t, \cdots, c_n\}$ are observed discourse constituents and $L = \{l_1, \cdots, l_t, \cdots, l_n\}$ are hidden structure nodes; label $l_t \in \{1, 0\}$ denotes whether constituents c_t and c_{t+1} should be connected. Our model differs from standard linear-chain CRFs in that the score between adjacent hidden nodes is not calculated based on a transition matrix, but is learned from observations instead. Specifically,

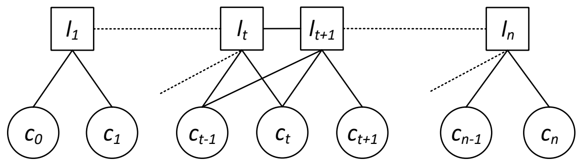


Figure 2: Intra-sentential structure CRF with pairwise modeling.

given a constituent sequence C, the probability of forming the structure sequence Y^{str} is written as:

$$p(Y^{str}|C) = \frac{1}{Z}\prod_{t=1}^{n} exp(u_t^{str} + b_{t,t+1}^{str}) \quad (1)$$

$$u_t^{str} = \theta(c_t, c_{t+1}, y_t) \quad (2)$$

$$b_{t,t+1}^{str} = \gamma(c_{t-1}, c_t, c_{t+1}, y_t, y_{t+1}) \quad (3)$$

where the u_t^{str} represents the unary potential score of structure node $l_t = y_t$, depending on constituents c_t and c_{t+1}. The binary potential score $b_{t,t+1}^{str}$ for $s_t = y_t, s_{t+1} = y_{t+1}$ is calculated based on c_{t-1}, c_t and c_{t+1}. The binary potential provides information for discriminating between $s_t = 0, s_{t+1} = 1$ and $s_t = 1, s_{t+1} = 0$. Also, we impose the constraint that one constituent can be merged with at most one other adjacent constituent.

Figure 3 depicts the relation CRF for intra-sentential parsing. Similar to the structure CRF, $C = \{c_0, \cdots, c_t, \cdots, c_n\}$ are observed constituents and $R = \{r_1, \cdots, r_t, \cdots, r_n\}$ hidden nodes, corresponding to discourse relations. The CRF will assign a relation to a non-leaf constituent c_t based on its right and left children, $c_{t,L}$ and $c_{t,R}$, respectively. If a constituent is a single EDU, we force its hidden node to be the special label *LEAF*, whereas hidden nodes for constituents which are the product of merging cannot be *LEAF*. Given a sequence of constituents C, the probability of the relation label sequence Y^{rel} can be written as:

$$p(Y^{rel}|C) = \frac{1}{Z}\prod_{t=1}^{n} exp(u_t^{rel} + b_{t,t+1}^{rel}) \quad (4)$$

$$u_t^{rel} = \theta(c_t, y_t) \quad (5)$$

$$b_{t,t+1}^{rel} = \gamma(c_t, c_{t+1}, y_t, y_{t+1}) \quad (6)$$

4.2 Intra-sentential Feature Learning

Instead of adopting a high order parsing model, we use neural networks to capture contextual information and recover the meaning of discourse constituents. Aside from modeling long distance dependencies, our representation learning process

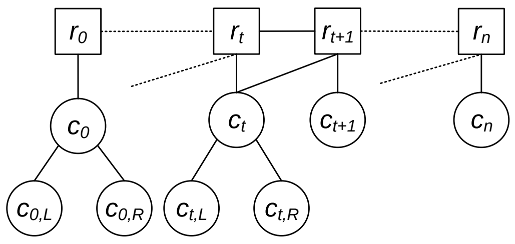


Figure 3: Intra-sentential relation CRF with pairwise modeling.

alleviates the need for elaborate feature engineering and selection. Our approach is based on LSTMs (Hochreiter and Schmidhuber, 1997) which have recently emerged as a popular architecture for modeling sequences and have been successfully applied to a variety of tasks ranging from machine translation (Sutskever et al., 2014), to speech recognition (Graves et al., 2013), and image description generation (Vinyals et al., 2015b). LSTMs have also been incorporated into syntactic parsing in a variety of ways (Vinyals et al. 2015a; Kiperwasser and Goldberg 2016; Dyer et al. 2015, *inter alia*). Of particular relevance to this work is LSTM-minus, a method for learning embeddings of text spans, which has achieved competitive performance in both dependency and constituency parsing (Wang and Chang, 2016; Cross and Huang, 2016). We describe below how we extend this method which is based on subtraction between LSTM hidden vectors to discourse parsing.

We represent each sentence as a sequence of word embeddings $[\boldsymbol{w}_{sos}, \boldsymbol{w}_1, \cdots, \boldsymbol{w}_i, \cdots, \boldsymbol{w}_n, \boldsymbol{w}_{eos}]$ and insert a special embedding w_E to indicate the boundaries of EDUs. We run a bidirectional LSTM over the sentence and obtain the output vector sequence $[\boldsymbol{h}_0, \cdots, \boldsymbol{h}_i, \cdots, \boldsymbol{h}_t]$, where $\boldsymbol{h}_i = [\overrightarrow{\boldsymbol{h}}_i, \overleftarrow{\boldsymbol{h}}_i]$ is the output vector for the i^{th} word, and $\overrightarrow{\boldsymbol{h}}_i$ and $\overleftarrow{\boldsymbol{h}}_i$ are the output vectors from the forward and backward directions, respectively. We represent a constituent c from position a to b with a span vector $\boldsymbol{sp}$ which is the concatenation of the vector differences $\overrightarrow{\boldsymbol{h}}_{b+1} - \overrightarrow{\boldsymbol{h}}_a$ and $\overleftarrow{\boldsymbol{h}}_{a-1} - \overleftarrow{\boldsymbol{h}}_b$:

$$\boldsymbol{sp} = [\overrightarrow{\boldsymbol{h}}_{b+1} - \overrightarrow{\boldsymbol{h}}_a, \overleftarrow{\boldsymbol{h}}_{a-1} - \overleftarrow{\boldsymbol{h}}_b] \quad (7)$$

As illustrated in Figure 4, spans are represented using output from both backward and forward LSTM components. Intuitively, this allows to obtain representations for EDUs and larger constituents in context, as embeddings are learned based on in-

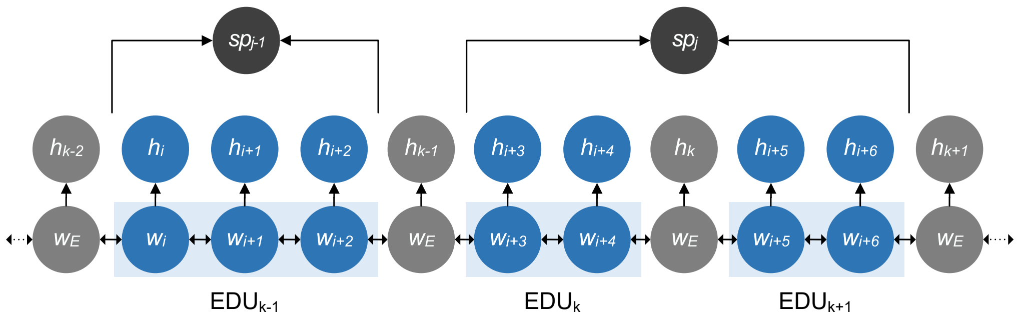


Figure 4: Modeling discourse constituents by LSTM-minus features. The feature vector $\boldsymbol{sp}_j$ for a constituent covering EDU$_k$ and EDU$_{k+1}$ is $[\vec{\boldsymbol{h}}_{k+1} - \vec{\boldsymbol{h}}_{i+3}, \overleftarrow{\boldsymbol{h}}_{k-1} - \overleftarrow{\boldsymbol{h}}_{i+6}]$. Blue nodes indicate word embeddings and LSTM outputs for words, while gray nodes represent EDU separators. Black nodes are learned LSTM-minus features for constituents.

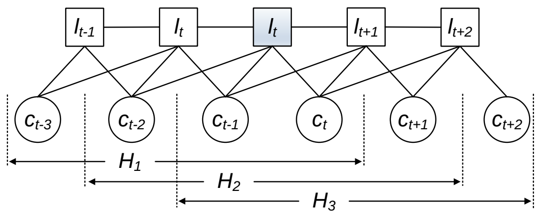


Figure 5: Multi-sentential structure CRF with sliding-window. The window size is 3, H_1, H_2, and H_3 denote the windows for predicting s_t, which is highlighted by the shaded rectangle.

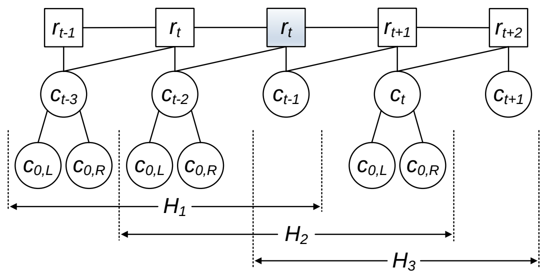


Figure 6: Multi-sentential relation CRF with sliding-window. The window size is 3, H_1, H_2 and H_3 denote the windows for predicting r_t, which highlighted by the shaded rectangle.

formation from the span itself and the words surrounding it.

The structure CRF model calculates the unary potential score u_t^{str} and the binary potential score $b_{t,t+1}^{str}$ based on the span vector as follows:

$$\boldsymbol{u}_t^{str} = \boldsymbol{W}_u^{str}[\boldsymbol{sp}_j, \boldsymbol{sp}_{j+1}] \quad (8)$$

$$\boldsymbol{b}_{t,t+1}^{str} = \boldsymbol{W}_b^{str}[\boldsymbol{sp}_{j-1}, \boldsymbol{sp}_j, \boldsymbol{sp}_{j+1}] \quad (9)$$

where $\boldsymbol{sp}_j$ is the span vector for the j^{th} constituent in the current sequence; and $\boldsymbol{W}_u^{str} \in \mathbb{R}^{d\times 2}, \boldsymbol{W}_b^{str} \in \mathbb{R}^{d\times 4}$ are weight matrices. $\boldsymbol{b}_t^{str}$ is reshaped into a 2×2 matrix, where the $(i, j)^{th}$ entry indicates the score of the transition from label i to label j between constituent c_t and c_{t+1}.

Analogously, unary and binary potential scores for the relation CRF are calculated as:

$$\boldsymbol{u}_t^{rel} = \boldsymbol{W}_u^{rel}[\boldsymbol{sp}_{j,L}, \boldsymbol{sp}_{j,R}] \quad (10)$$

$$\boldsymbol{b}_{t,t+1}^{rel} = \boldsymbol{W}_b^{rel}[\boldsymbol{sp}_{j,L}, \boldsymbol{sp}_{j,R}, \boldsymbol{sp}_{j+1,L}, \boldsymbol{sp}_{j+1,R}] \quad (11)$$

where $\boldsymbol{sp}_{j,L}$ and $\boldsymbol{sp}_{j,R}$ are span vector representations for the left and right child of the j^{th} constituent; $\boldsymbol{W}_u^{rel} \in \mathbb{R}^{d\times n_r}, \boldsymbol{W}_b^{rel} \in \mathbb{R}^{d\times n_r^2}$, and n_r is the number of discourse relations; $\boldsymbol{b}_t^{rel}$ is also reshaped into a $n_r \times n_r$ matrix. For a constituent that is a single EDU, $\boldsymbol{sp}_{j,R}$ and $\boldsymbol{sp}_{j+1,R}$ are special vectors $\boldsymbol{w}_{LEAF_L}$ and $\boldsymbol{w}_{LEAF_R}$.

4.3 Multi-sentential Parser

The multi-sentential discourse parser treats sentences as the smallest possible discourse units, following a process similar to the intra-sentential model. Unfortunately, it is practically unfeasible to model all constituents with one CRF when processing entire documents. The forward-backward algorithm for calculating the CRF normalization factor on a sequence with T units has time complexity $O(TM^2)$, where M is the number of labels, leading to $O(T^2M^2)$ time for parsing a document.

We therefore modify the two CRFs into sliding-window versions. Specifically, at each level, when decoding a constituent sequence, for each hidden structure node l_i or relation node r_i, we find all windows of constituents that contain the hidden node, and set the hidden node's label according to the window with the maximum joint probability.

We present the modified sliding-window CRFs in Figure 5 (structure) and Figure 6 (relation).

4.4 Multi-sentential Feature Learning

For multi-sentential parsing, we also use the LSTM-minus method to model constituents, however, the minimum units here are sentences rather than EDUs. We represent individual sentences with the same bidirectional LSTM used for intra-sentential parsing. For sentence s_i, the LSTM output vector of the first token $\boldsymbol{h}_0$ and the last token $\boldsymbol{h}_n$ are taken to form the sentence representation $\boldsymbol{v}_i$:

$$\boldsymbol{v}_i = [\boldsymbol{h}_0, \boldsymbol{h}_n] \tag{12}$$

In order to capture additional contextual information, we then build another bidirectional LSTM over a sequence of sentence vectors. We learn representations for constituents (aka text spans within a document) with the LSTM-minus method and use $\boldsymbol{fl}$ to denote the resulting vectors. Although this flat representation is relatively straightforward to obtain, it is perhaps overly simplistic for modeling documents where the number of sentences can be relative large. Moreover, it is not clear that it is discriminating enough for modeling relations between constituents. Such relations follow structural regularities (e.g., they can be asymmetric or symmetric, left- or right-branching) which cannot be captured when adopting a sequence-based document view.

To inject structural knowledge in our representation, we also model constituents as subtrees with recursive neural networks (Socher et al., 2012). The latter operate over tree structures which we obtain during training from the RST Discourse Treebank (Carlson et al., 2003). The representation for each parent is computed based on its children iteratively, in a bottom-up fashion. More formally, let vectors $\boldsymbol{h}_L$ and $\boldsymbol{h}_R$ denote the left and right children of constituent c and dis their rhetorical relation. The vector for parent c is:

$$\boldsymbol{h}_c = \tanh(\boldsymbol{W}_{dis}[\boldsymbol{h}_L, \boldsymbol{h}_R] + \boldsymbol{b}_{dis}) \tag{13}$$

where $[\boldsymbol{h}_L, \boldsymbol{h}_R]$ is the concatenation of the children representations $\boldsymbol{h}_L \in \mathbb{R}^d$ and $\boldsymbol{h}_R \in \mathbb{R}^d$, $\boldsymbol{W}_{dis} \in \mathbb{R}^{2d \times d}$, and $\boldsymbol{b}_{dis} \in \mathbb{R}^d$ is the bias vector. We henceforth use the term tree vector $\boldsymbol{tr}$ to refer to the representation in Equation (13) since constituents are now subtrees in a document-wide discourse tree.

In multi-sentential parsing, the span vector $\boldsymbol{sp}$ now becomes the concatenation of the flat vector $\boldsymbol{fl}$ and the tree vector $\boldsymbol{tr}$:

$$\boldsymbol{sp} = [\boldsymbol{fl}, \boldsymbol{tr}] \tag{14}$$

The representation above, attempts to capture richer semantic features during the tree building process while benefiting from the robustness afforded by the flat LSTM-based text spans. An example of this representation is given in Figure 7.

Unary and binary potential scores for the structure and relation CRFs are calculated as in intra-sentential parsing (see Section 4.2).

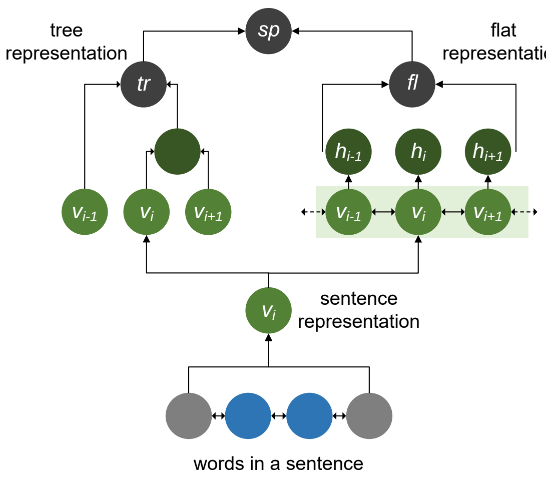


Figure 7: Document-level constituents with LSTM-minus features. Blue nodes are LSTM outputs for words, light green nodes represent vectors for sentences, dark green nodes are LSTM outputs for sentences, and black nodes are learned constituent representations.

4.5 Training

We first train the intra-sentential parser and use the learned LSTM as a component of the multi-sentential parser. During training, we maximize the log-likelihood of the correct label sequence Y given a constituent sequence C, which is $p(Y^{str}|C)$ for the structure CRFs and $p(Y^{rel}|C)$ for the relation CRFs. Stochastic gradient descent with momentum is used to update the parameters of the network. In our experiments, the momentum is set to 0.9 and the learning rate is 0.001. The LSTMs in our paper have one hidden layer.

5 Experimental Setup

In this section we present our experimental setup for assessing the performance of the discourse parser described above. We give details on the datasets we used, evaluation protocol, and model training.

Evaluation We evaluated our model on the RST Discourse Treebank (RST-DT; Carlson et al. 2003), which is partitioned into 347 documents for training and 38 documents for testing. Following previous work (Joty et al., 2013; Li et al., 2016), we converted non-binary relations into a cascade of right-branching binary relations.

Predicted RST-trees are typically evaluated by computing F1 against gold standard trees (Marcu, 2000). Evaluation metrics for RST-style discourse parsing include: (a) *span* (S) which measures whether the predicted subtrees match the goldstandard; (b) *nucleus* (N) which measures whether subtrees have the same nucleus as in the goldstandard and (c) *relation* (R) which measures whether discourse relations have been identified correctly. The three metrics are interdependent, errors on the span metric propagate to the nuclearity metric, and in turn to the relation metric. Following other RST-style discourse parsing systems (Hernault et al., 2010), we evaluate the relation metric using 18 coarse-grained relation classes, and with nuclearity attached, we have a total of 41 distinct relations.[1] Since EDU segmentation falls outside the scope of this work, we evaluate our system on gold-standard EDUs. Comparison systems are also assessed in the same setting.

Training Details Word embeddings were pretrained with the Gensim[2] implementation of word2vec (Mikolov et al., 2013) on the English GigaWord corpus (with case left intact). The dimensionality of the word embeddings was set to 50. Following Li et al. (2016), the embeddings were fine-tuned using a mapping matrix $\boldsymbol{W} \in \mathbb{R}^{50\times 50}$ trained with the following criterion:

$$\min_{\boldsymbol{W},\boldsymbol{b}} \|\boldsymbol{LT}_{tuned} - \boldsymbol{LT}_{pre}\boldsymbol{W} + \boldsymbol{b}\| \qquad (15)$$

where $\boldsymbol{LT}_{tuned}$, and $\boldsymbol{LT}_{pre}$ are lookup tables for fine-tuned and pre-trained word embeddings in the training set. Matrix W can be subsequently used to to estimate fine-tuned embeddings for words in the test set.

Tokenization, POS tagging and sentence splitting were performed using the Stanford CoreNLP toolkit (Manning et al., 2014). All neural network parameters were initialized randomly with Xavier's initialization (Glorot and Bengio, 2010). The hyper-parameters are tuned by cross-validation on the training set.

[1]For calculating the binary potential scores, 41 relations will lead to a large number of parameters; to avoid this, we only use 18 relations without nuclearity.

[2]`https://radimrehurek.com/gensim/`

CIDER	S	N	R
Tree Span	79.5	68.1	56.6
Flat Span (−minus)	82.7	69.3	55.6
Flat Span (+minus)	83.6	70.1	55.4
Tree + Flat Span (+minus)	83.6	71.1	57.3

Table 1: CIDER performance using different constituent representations (RST-DT test set).

Additional Features Most existing state-of-the-art systems rely heavily on handcrafted features (Hernault et al., 2010; Feng and Hirst, 2014; Joty et al., 2013) some of which have been also proved helpful in neural network models (Li et al., 2014, 2016). In our experiments, we use the following basic features which have been widely adopted in various discourse parsing models: (1) the first three words and the last two words of each constituent; (2) the POS tags of the first three words and the last two words of each constituent; (3) the number of EDUs; and (4) the number of tokens of each constituent. We concatenate these features with the constituent vectors learned by our neural networks, and train new CRF models.

6 Results

In this paper we have presented two views for modeling discourse constituents, namely as trees or sequences. We experimentally assessed whether these two views are overlapping or complementary. Table 1 reports the performance of our parser which we call CIDER (as a shorthand for **C**ontextually **I**nformed **D**iscourse Pars**er**) without the additional features introduced in Section 5. The first row presents a version of CIDER based solely on tree span representations. In the second row, CIDER uses flat representations without LSTM minus features. Specifically, each constituent is represented as the average of the LSTM outputs within it. In the third row, CIDER's representations are computed with the LSTM minus method, while the fourth row shows results for the full system.

As can be seen, on the span (S) metric, CIDER with flat span representations is much better than CIDER with tree span representations; on the nuclearity (N) metric tree representations are still inferior to flat representations but the performance

Discourse Parsers	S	N	R	Speed
Ji and Eisenstein (2014)	82.1	71.1	61.6	0.21
Feng and Hirst (2014)	85.7	71.0	58.2	9.88
Heilman and Sagae (2015)	83.5	68.1	55.1	0.40
CIDER (−AF)	83.6	71.1	57.3	3.80
CIDER (+AF)	85.0	71.1	59.0	3.80
Li et al. (2014)	84.0	70.8	58.6	26.00
Li et al. (2016)	85.8	71.7	58.9	–
Human	88.7	77.7	65.8	–

Table 2: Comparison with state-of-the-art systems (RST-DT test set). Speed indicates the average number seconds taken to parse a document.

gap is narrower, whereas on the the relation (R) metric, tree span representations are superior. We believe this can be explained by the fact that relation identification relies on more semantic information while span identification relies on more shallow features, like the beginning and the end of the span (Ji and Eisenstein, 2014). Span features based on the LSTM minus method bring improvements over vanilla LSTM representations on the span and nuclearity metrics. Perhaps unsurprisingly, the combination of span and tree representations achieves the best results overall.

In Table 2, we compare our system with several state-of-the-art discourse parsers, which can be classified in two groups depending on their time complexity. Linear-time systems (first block in the table) include two transition-based parsers (Ji and Eisenstein, 2014; Heilman and Sagae, 2015) and one CRF-based parser (Feng and Hirst, 2014), whereas cubic-time parsers (second block) include two neural network models (Li, Li, and Hovy, 2014; Li, Li, and Chang, 2016). CIDER falls in the first group as it is a liner-time parser, while it shares with parsers in the second group the use of neural architectures for automated feature extraction. We report CIDER scores with and without the additional features (AF) discussed in the previous section. As an upper bound, we also report inter-annotator agreement on the discourse parsing task (last row in the table).

Amongst linear-time systems, our parser achieves comparable results on the span and relation metric, and best performance on the nuclearity metric. Note that the three metrics evaluate different aspects of a discourse parser, and CIDER achieves the most balanced results across all metrics. As far as other comparison systems are concerned, Ji and Eisenstein (2014) employ a shift-reduce discourse parser. They represent EDUs with word-count vectors and use a projection matrix to combine them into text spans. A support vector machine classifier is used to decide the actions of the parser. Heilman and Sagae (2015) also adopt a shift-reduce approach and use multi-class logistic regression to select the best parsing action. Their classifier considers a variety of lexical, syntactic, and positional features. Feng and Hirst's (2014) system is closest to ours in their use of linear-chain CRFs, but their features are mainly extracted from local constituents. Furthermore, they adopt a post-editing method which modifies the discourse trees their parser creates with height features.

With regard to previously proposed cubic-time systems, CIDER outperforms Li et al. (2014) across all metrics. Their CYK-based parser adopts a recursive deep model for composing EDUs hierarchically together with several additional features to boost performance. CIDER performs slightly worse on span and nuclearity compared to Li et al. (2016), but is better at identifying relations. Their system uses an attention-based hierarchical neural network for modeling text spans and a tensor-based transformation for combining two spans. A CKY-like algorithm is used to generate the discourse tree structure. In comparison, CIDER is conceptually simpler, and more efficient.

We used paired bootstrap re-sampling (Efron and Tibshirani, 1993) to assess whether differences in performance are statistically significant. CIDER is significantly better than Feng and Hirst's 2014 system on the relation metric ($p < 0.05$); it is also significantly better ($p < 0.05$) than Heilman and Sagae (2015) on all three metrics and better than Ji and Eisenstein (2014) on the span metric. Compared to Li et al. (2014), CIDER is significantly better on the span and relation metrics ($p < 0.05$). Unfortunately, we cannot perform significance tests against Li et al. (2016) as we do not have access to the output of their system.

We also evaluated the speed of CIDER and comparison discourse parsers on a platform with Intel Core-i5-7200U CPU at 2.50GHz. We report the average number of seconds taken to parse a document in the RST-DT test set. The times shown in Table 2 do not include pre-processing, which for CIDER is only part-of-speech tagging, whereas all other linear-time systems rely on a syntactic parser. As can be seen CIDER is quite efficient compared to related systems (Feng and Hirst, 2014;

Heilman and Sagae, 2015) whilst requiring less feature engineering.

7 Conclusions

In this paper we described CIDER, a simple and efficient discourse parser which adopts a two-stage parsing strategy, whilst exploiting a more global feature space. We proposed a novel way to learn contextually informed representations of constituents with the LSTM minus method, at the sentence and document level. We also demonstrated that flat representations of text spans can be usefully complemented with tree-based ones leading to a more accurate characterization of discourse relations. Experimental results showed that CIDER performs on par with the state of the art (Li et al., 2016), despite the greedy parsing algorithm and relatively simple neural architecture. In the future, we would like to improve parsing accuracy by leveraging unlabeled text rather than relying exclusively on human annotated training data.

Acknowledgments The authors gratefully acknowledge the support of the European Research Council (award number 681760).

References

Jason Baldridge and Alex Lascarides. 2005. Probabilistic head-driven parsing for discourse structure. In *Proceedings of the CoNLL conference*. pages 96–103.

Lynn Carlson, Daniel Marcu, and Mary Ellen Okurowski. 2003. Building a discourse-tagged corpus in the framework of rhetorical structure theory. In *Current and new directions in discourse and dialogue*, Springer, pages 85–112.

Joyce Y. Chai and Rong Jin. 2004. Discourse structure for context question answering. In *HLT-NAACL 2004: Workshop on Pragmatics of Question Answering*. pages 23–30.

James Cross and Liang Huang. 2016. Span-based constituency parsing with a structure-label system and provably optimal dynamic oracles. In *Proceedings of the EMNLP conference*. pages 1–11.

Chris Dyer, Miguel Ballesteros, Wang Ling, Austin Matthews, and Noah A. Smith. 2015. Transition-based dependency parsing with stack long short-term memory. In *Proceedings of the ACL conference*. pages 334–343.

Bradley Efron and Robert J. Tibshirani. 1993. *An Introduction ot the Bootstrap*. Chapman & Hall, New York, NY.

Vanessa Wei Feng and Graeme Hirst. 2012. Text-level discourse parsing with rich linguistic features. In *Proceedings of the ACL conference*. pages 60–68.

Vanessa Wei Feng and Graeme Hirst. 2014. A linear-time bottom-up discourse parser with constraints and post-editing. In *Proceedings of the ACL conference*. pages 511–521.

Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics*. volume 9, pages 249–256.

Alex Graves, Jaitly Navdeep, and A-R Mohamed. 2013. Hybrid speech recognition with deep bidirectional lstm. In *IEEE Workshop on Automatic Speech Recognition and Understanding*. pages 293–278.

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

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

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9(8):1735–1780.

Yangfeng Ji and Jacob Eisenstein. 2014. Representation learning for text-level discourse parsing. In *Proceedings of the ACL conference*. pages 13–24.

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 ACL conference*. pages 486–496.

Eliyahu Kiperwasser and Yoav Goldberg. 2016. Simple and accurate dependency parsing using bidirectional lstm feature representations. *Transactions of the Association for Computational Linguistics* 4:313–327.

Huong LeThanh, Geetha Abeysinghe, and Christian Huyck. 2004. Generating discourse structures for written texts. In *Proceedings of the 20th international conference on Computational Linguistics*. page 329.

Jiwei Li, Rumeng Li, and Eduard Hovy. 2014. Recursive deep models for discourse parsing. In *Proceedings of the EMNLP conference*. pages 2061–2069.

Qi Li, Tianshi Li, and Baobao Chang. 2016. Discourse parsing with attention-based hierarchical neural networks. In *Proceedings of the EMNLP conference*. pages 362–371.

William C Mann and Sandra A Thompson. 1988. Rhetorical structure theory: Toward a functional theory of text organization. *Text-Interdisciplinary Journal for the Study of Discourse* 8(3):243–281.

Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of the ACL conference: System Demonstrations*. pages 55–60.

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

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* .

Rashmi Prasad, Nikhil Dinesh, Alan Lee, Eleni Miltsakaki, Livio Robaldo, Aravind K Joshi, and Bonnie L Webber. 2008. The Penn discourse treebank 2.0. In *Proceedings of LREC*.

Kenji Sagae. 2009. Analysis of discourse structure with syntactic dependencies and data-driven shift-reduce parsing. In *Proceedings of the 11th International Conference on Parsing Technologies*. pages 81–84.

Richard Socher, Brody Huval, Christopher D. Manning, and Andrew Y. Ng. 2012. Semantic compositionality through recursive matrix-vector spaces. In *Proceedings of the EMNLP conference*. pages 1201–1211.

Swapna Somasundaran. 2010. *Discourse-level Relation for Opinion Analysis*. Ph.D. thesis, University of Pittsburgh.

Radu Soricut and Daniel Marcu. 2003. Sentence level discourse parsing using syntactic and lexical information. In *Proceedings of the NAACL conference*. pages 149–156.

Rajen Subba and Barbara Di Eugenio. 2009. An effective discourse parser that uses rich linguistic information. In *Proceedings of the ACL conference*. pages 566–574.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Proceedings of the NIPS conference*. pages 3104–3112.

Oriol Vinyals, Łukasz Kaiser, Terry Koo, Slav Petrov, Ilya Sutskever, and Geoffrey Hinton. 2015a. Grammar as a foreign language. In *Proceedings of the NIPS conference*. pages 2773–2781.

Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. 2015b. Show and tell: A neural image caption generator. In *Proceedings of the CVPR conference*. pages 3156–3164.

Wenhui Wang and Baobao Chang. 2016. Graph-based dependency parsing with bidirectional lstm. In *Proceedings of the ACL conference*. pages 2306–2315.

Yasuhisa Yoshida, Jun Suzuki, Tsutomu Hirao, and Masaaki Nagata. 2014. Dependency-based discourse parser for single-document summarization. In *Proceedings of the EMNLP conference*. pages 1834–1839.