Representation Learning for Text-level Discourse Parsing

Yangfeng Ji
School of Interactive Computing
Georgia Institute of Technology
jiyfeng@gatech.edu

Jacob Eisenstein
School of Interactive Computing
Georgia Institute of Technology
jacobe@gatech.edu

Abstract

Text-level discourse parsing is notoriously difficult, as distinctions between discourse relations require subtle semantic judgments that are not easily captured using standard features. In this paper, we present a representation learning approach, in which we transform surface features into a latent space that facilitates RST discourse parsing. By combining the machinery of large-margin transition-based structured prediction with representation learning, our method jointly learns to parse discourse while at the same time learning a discourse-driven projection of surface features. The resulting shift-reduce discourse parser obtains substantial improvements over the previous state-of-the-art in predicting relations and nuclearity on the RST Treebank.

1 Introduction

Discourse structure describes the high-level organization of text or speech. It is central to a number of high-impact applications, such as text summarization (Louis et al., 2010), sentiment analysis (Voll and Taboada, 2007; Somasundaran et al., 2009), question answering (Ferrucci et al., 2010), and automatic evaluation of student writing (Miltsakaki and Kukich, 2004; Burstein et al., 2013). Hierarchical discourse representations such as Rhetorical Structure Theory (RST) are particularly useful because of the computational applicability of tree-shaped discourse structures (Taboada and Mann, 2006), as shown in Figure 1.

Unfortunately, the performance of discourse parsing is still relatively weak: the state-of-the-art F-measure for text-level relation detection in the RST Treebank is only slightly above 55% (Joty et al., 2013). While recent work has introduced increasingly powerful features (Feng and Hirst, 2012) and inference techniques (Joty et al., 2013), discourse relations remain hard to detect, due in part to a long tail of "alternative lexicalizations" that can be used to realize each relation (Prasad et al., 2010). Surface and syntactic features are not capable of capturing what are fundamentally semantic distinctions, particularly in the face of relatively small annotated training sets.

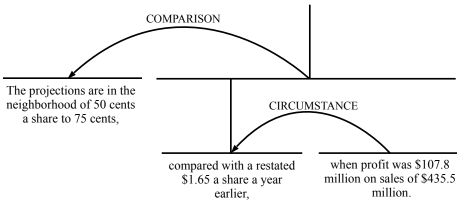

COMPARISON
The projections are in the neighborhood of 50 cents a share to 75 cents,
CIRCUMSTANCE
compared with a restated $1.65 a share a year earlier,
when profit was $107.8 million on sales of $435.5 million.

Figure 1: An example of RST discourse structure.

In this paper, we present a representation learning approach to discourse parsing. The core idea of our work is to learn a transformation from a bag-of-words surface representation into a latent space in which discourse relations are easily identifiable. The latent representation for each discourse unit can be viewed as a discriminatively-trained vector-space representation of its meaning. Alternatively, our approach can be seen as a nonlinear learning algorithm for incremental structure prediction, which overcomes feature sparsity through effective parameter tying. We consider several alternative methods for transforming the original features, corresponding to different ideas of the meaning and role of the latent representation.

Our method is implemented as a shift-reduce discourse parser (Marcu, 1999; Sagae, 2009). Learning is performed as large-margin transition-based structure prediction (Taskar et al., 2003), while at the same time jointly learning to project the surface representation into latent space. The resulting system strongly outperforms the prior state-of-the-art at labeled F-measure, obtaining raw improvements of roughly 6% on relation labels and 2.5% on nuclearity. In addition, we show that the latent representation coheres well with the characterization of discourse connectives in the Penn Discourse Treebank (Prasad et al., 2008).

2 Model

The core idea of this paper is to project lexical features into a latent space that facilitates discourse parsing. In this way, we can capture the meaning of each discourse unit, without suffering from the very high dimensionality of a lexical representation. While such feature learning approaches have proven to increase robustness for parsing, POS tagging, and NER (Miller et al., 2004; Koo et al., 2008; Turian et al., 2010), they would seem to have an especially promising role for discourse, where training data is relatively sparse and ambiguity is considerable. Prasad et al. (2010) show that there is a long tail of alternative lexicalizations for discourse relations in the Penn Discourse Treebank, posing obvious challenges for approaches based on directly matching lexical features observed in the training data.

Based on this observation, our goal is to learn a function that transforms lexical features into a much lower-dimensional latent representation, while simultaneously learning to predict discourse structure based on this latent representation. In this paper, we consider a simple transformation function, linear projection. Thus, we name the approach DPLP: Discourse Parsing from Linear Projection. We apply transition-based (incremental) structured prediction to obtain a discourse parse, training a predictor to make the correct incremental moves to match the annotations of training data in the RST Treebank. This supervision signal is then used to learn both the weights and the projection matrix in a large-margin framework.

2.1 Shift-reduce discourse parsing

We construct RST Trees using shift-reduce parsing, as first proposed by Marcu (1999). At each point in the parsing process, we maintain a stack and a queue; initially the stack is empty and the first elementary discourse unit (EDU) in the document is at the front of the queue.[1] The parser can then choose either to *shift* the front of the queue onto the top of the stack, or to *reduce* the top two elements on the stack in a discourse relation. The reduction operation must choose both the type of relation and which element will be the nucleus. So, overall there are multiple reduce operations with specific relation types and nucleus positions. Shift-reduce parsing can be learned as a classification task, where the classifier uses features of the elements in the stack and queue to decide what move to take. Previous work has employed decision trees (Marcu, 1999) and the averaged perceptron (Collins and Roark, 2004; Sagae, 2009) for this purpose. Instead, we employ a large-margin classifier, because we can compute derivatives of the margin-based objective function with respect to both the classifier weights as well as the projection matrix.

Notation	Explanation
$\mathcal{V}$	Vocabulary for surface features
V	Size of $\mathcal{V}$
K	Dimension of latent space
$\mathbf{w}_m$	Classification weights for class m
C	Total number of classes, which correspond to possible shift-reduce operations
$\mathbf{A}$	Parameter of the representation function (also the projection matrix in the linear representation function)
$\mathbf{v}_i$	Word count vector of discourse unit i
$\mathbf{v}$	Vertical concatenation of word count vectors for the three discourse units currently being considered by the parser
λ	Regularization for classification weights
τ	Regularization for projection matrix
ξ_i	Slack variable for sample i
$\eta_{i,m}$	Dual variable for sample i and class m
α_t	Learning rate at iteration t

Table 1: Summary of mathematical notation

2.2 Discourse parsing with projected features

More formally, we denote the surface feature vocabulary $\mathcal{V}$, and represent each EDU as the numeric vector $\mathbf{v} \in \mathbb{N}^V$, where $V = \#|\mathcal{V}|$ and the n-th element of $\mathbf{v}$ is the count of the n-th surface feature in this EDU (see Table 1 for a summary of notation). During shift-reduce parsing, we consider features of three EDUs:[2] the top two elements on

[1] We do not address segmentation of text into elementary discourse units in this paper. Standard classification-based approaches can achieve a segmentation F-measure of 94% (Hernault et al., 2010); a more complex reranking model does slightly better, at 95% F-Measure with automatically-generated parse trees, and 96.6% with gold annotated trees (Xuan Bach et al., 2012). Human agreement reaches 98% F-Measure.

[2] After applying a reduce operation, the stack will include a span that contains multiple EDUs. We follow the *strong*

the stack ($\mathbf{v}_1$ and $\mathbf{v}_2$), and the front of the queue ($\mathbf{v}_3$). The vertical concatenation of these vectors is denoted $\mathbf{v} = [\mathbf{v}_1; \mathbf{v}_2; \mathbf{v}_3]$. In general, we can formulate the decision function for the multi-class shift-reduce classifier as

$$\hat{m} = \operatorname*{arg\,max}_{m \in \{1,\dots,C\}} \mathbf{w}_m^\top \mathbf{f}(\mathbf{v}; \mathbf{A}) \tag{1}$$

where $\mathbf{w}_m$ is the weight for the m-th class and $\mathbf{f}(\mathbf{v}; \mathbf{A})$ is the *representation function* parametrized by $\mathbf{A}$. The score for class m (in our case, the value of taking the m-th shift-reduce operation) is computed by the inner product $\mathbf{w}_m^\top \mathbf{f}(\mathbf{v}; \mathbf{A})$. The specific shift-reduce operation is chosen by maximizing the decision value in Equation 1.

The representation function $\mathbf{f}(\mathbf{v}; \mathbf{A})$ can be defined in any form; for example, it could be a non-linear function defined by a neural network model parametrized by $\mathbf{A}$. We focus on the linear projection,

$$\mathbf{f}(\mathbf{v}; \mathbf{A}) = \mathbf{A}\mathbf{v}, \tag{2}$$

where $\mathbf{A} \in \mathbb{R}^{K \times 3V}$ is projects the surface representation $\mathbf{v}$ of three EDUs into a latent space of size $K \ll V$.

Note that by setting $\tilde{\mathbf{w}}_m^\top = \mathbf{w}_m^\top \mathbf{A}$, the decision scoring function can be rewritten as $\tilde{\mathbf{w}}_m^\top \mathbf{v}$, which is linear in the original surface features. Therefore, the expressiveness of DPLP is identical to a linear separator in the original feature space. However, the learning problem is considerably different. If there are C total classes (possible shift-reduce operations), then a linear classifier must learn $3VC$ parameters, while DPLP must learn $(3V + C)K$ parameters, which will be smaller under the assumption that $K < C \ll V$. This can be seen as a form of *parameter tying* on the linear weights $\tilde{\mathbf{w}}_m$, which allows statistical strength to be shared across training instances. We will consider special cases of $\mathbf{A}$ that reduce the parameter space still further.

2.3 Special forms of the projection matrix

We consider three different constructions for the projection matrix $\mathbf{A}$.

- *General form*: In the general case, we place no special constraint on the form of $\mathbf{A}$.

$$\mathbf{f}(\mathbf{v}; \mathbf{A}) = \mathbf{A} \begin{bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \\ \mathbf{v}_3 \end{bmatrix} \tag{3}$$

  This form is shown in Figure 2(a).

- *Concatenation form*: In the concatenation form, we choose a block structure for $\mathbf{A}$, in which a single projection matrix $\mathbf{B}$ is applied to each EDU:

$$\mathbf{f}(\mathbf{v}; \mathbf{A}) = \begin{bmatrix} \mathbf{B} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{B} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{B} \end{bmatrix} \begin{bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \\ \mathbf{v}_3 \end{bmatrix} \tag{4}$$

  In this form, we transform the representation of each EDU separately, but do not attempt to represent interrelationships between the EDUs in the latent space. The number of parameters in $\mathbf{A}$ is $\frac{1}{3}KV$. Then, the total number of parameters, including the decision weights $\{\mathbf{w}_m\}$, in this form is $(\frac{V}{3} + C)K$.

- *Difference form*. In the difference form, we explicitly represent the *differences* between adjacent EDUs, by constructing $\mathbf{A}$ as a block difference matrix,

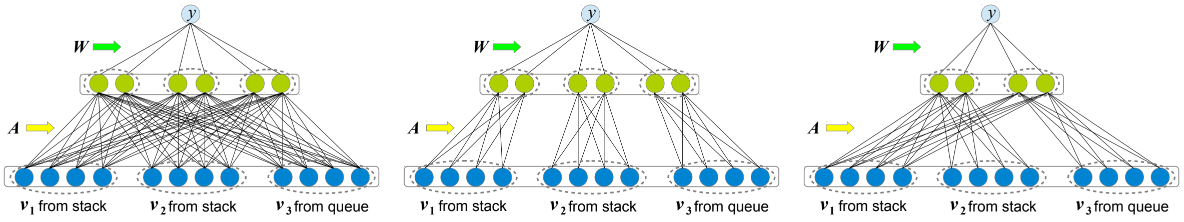
$$\mathbf{f}(\mathbf{v}; \mathbf{A}) = \begin{bmatrix} \mathbf{C} & -\mathbf{C} & \mathbf{0} \\ \mathbf{C} & \mathbf{0} & -\mathbf{C} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \\ \mathbf{v}_3 \end{bmatrix}, \tag{5}$$

  The result of this projection is that the latent representation has the form $[\mathbf{C}(\mathbf{v}_1 - \mathbf{v}_2); \mathbf{C}(\mathbf{v}_1 - \mathbf{v}_3)]$, representing the difference between the top two EDUs on the stack, and between the top EDU on the stack and the first EDU in the queue. This is intended to capture semantic similarity, so that reductions between related EDUs will be preferred. Similarly, the total number of parameters to estimate in this form is $(V + 2C)\frac{K}{3}$.

3 Large-Margin Learning Framework

We apply a large margin structure prediction approach to train the model. There are two parameters that need to be learned: the classification weights $\{\mathbf{w}_m\}$, and the projection matrix $\mathbf{A}$. As we will see, it is possible to learn $\{\mathbf{w}_m\}$ using standard support vector machine (SVM) training (holding $\mathbf{A}$ fixed), and then make a simple gradient-based update to $\mathbf{A}$ (holding $\{\mathbf{w}_m\}$ fixed). By interleaving these two operations, we arrive at a saddle point of the objective function.

compositionality criterion of Marcu (1996) and consider only the nuclear EDU of the span. Later work may explore the composition of features between the nucleus and satellite.



(a) General form (b) Concatenation form (c) Difference form

Figure 2: Decision problem with different representation functions

Specifically, we formulate the following constrained optimization problem,

$$\begin{aligned} &\min_{\{\mathbf{w}_{1:C},\xi_{1:l},\mathbf{A}\}} \frac{\lambda}{2}\sum_{m=1}^{C}\|\mathbf{w}_m\|_2^2 + \sum_{i=1}^{l}\xi_i + \frac{\tau}{2}\|\mathbf{A}\|_F^2 \\ &\text{s.t. } (\mathbf{w}_{y_i} - \mathbf{w}_m)^\top \mathbf{f}(\mathbf{v}_i;\mathbf{A}) \geq 1 - \delta_{y_i=m} - \xi_i, \\ &\qquad \forall\, i, m \end{aligned} \tag{6}$$

where $m \in \{1,\ldots,C\}$ is the index of the shift-reduce decision taken by the classifier (e.g., SHIFT, REDUCE-CONTRAST-RIGHT, etc), $i \in \{1,\cdots,l\}$ is the index of the training sample, and $\mathbf{w}_m$ is the vector of classification weights for class m. The slack variables ξ_i permit the margin constraint to be violated in exchange for a penalty, and the delta function $\delta_{y_i=m}$ is unity if $y_i = m$, and zero otherwise.

As is standard in the multi-class linear SVM (Crammer and Singer, 2001), we can solve the problem defined in Equation 6 via Lagrangian optimization:

$$\begin{aligned} &\mathcal{L}(\{\mathbf{w}_{1:C},\xi_{1:l},\mathbf{A},\eta_{1:l,1:C}\}) = \\ &\quad \frac{\lambda}{2}\sum_{m=1}^{C}\|\mathbf{w}_m\|_2^2 + \sum_{i=1}^{l}\xi_i + \frac{\tau}{2}\|\mathbf{A}\|_F^2 \\ &\quad + \sum_{i,m}\eta_{i,m}\Big\{(\mathbf{w}_m^\top - \mathbf{w}_{y_i}^\top)\mathbf{f}(\mathbf{v}_i;\mathbf{A}) + 1 - \delta_{y_i=m} - \xi_i\Big\} \\ &\quad \text{s.t. } \eta_{i,m} \geq 0\ \forall i, m \end{aligned} \tag{7}$$

Then, to optimize $\mathcal{L}$, we need to find a saddle point, which would be the minimum for the variables $\{\mathbf{w}_{1:C},\xi_{1:l}\}$ and the projection matrix $\mathbf{A}$, and the maximum for the dual variables $\{\eta_{1:l,1:C}\}$.

If $\mathbf{A}$ is fixed, then the optimization problem is equivalent to a standard multi-class SVM, in the transformed feature space $\mathbf{f}(\mathbf{v}_i;\mathbf{A})$. We can obtain the weights $\{\mathbf{w}_{1:C}\}$ and dual variables $\{\eta_{1:l,1:C}\}$ from a standard dual-form SVM solver. We then update $\mathbf{A}$, recompute $\{\mathbf{w}_{1:C}\}$ and $\{\eta_{1:l,1:C}\}$, and iterate until convergence. This iterative procedure is similar to the latent variable structural SVM (Yu and Joachims, 2009), although the specific details of our learning algorithm are different.

3.1 Learning Projection Matrix A

We update $\mathbf{A}$ while holding fixed the weights and dual variables. The derivative of $\mathcal{L}$ with respect to $\mathbf{A}$ is

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial \mathbf{A}} &= \tau\mathbf{A} + \sum_{i,m}\eta_{i,m}(\mathbf{w}_m^\top - \mathbf{w}_{y_i}^\top)\frac{\partial \mathbf{f}(\mathbf{v}_i;\mathbf{A})}{\partial \mathbf{A}} \\ &= \tau\mathbf{A} + \sum_{i,m}\eta_{i,m}(\mathbf{w}_m - \mathbf{w}_{y_i})\mathbf{v}_i{}^\top \end{aligned} \tag{8}$$

Setting $\frac{\partial \mathcal{L}}{\partial \mathbf{A}} = 0$, we have the closed-form solution,

$$\begin{aligned} \mathbf{A} &= \frac{1}{\tau}\sum_{i,m}\eta_{i,m}(\mathbf{w}_m - \mathbf{w}_{y_i})\mathbf{v}_i{}^\top \\ &= \frac{1}{\tau}\sum_{i,j}(\mathbf{w}_{y_i} - \sum_{m}\eta_{i,m}\mathbf{w}_m)\mathbf{v}_i{}^\top, \end{aligned} \tag{9}$$

because the dual variables for each instance must sum to one, $\sum_m \eta_{i,m} = 1$.

Note that for a given i, the matrix $(\mathbf{w}_{y_i} - \sum_m \eta_{i,m}\mathbf{w}_m)\mathbf{v}_i{}^\top$ is of (at most) rank-1. Therefore, the solution of $\mathbf{A}$ can be viewed as the linear combination of a sequence of rank-1 matrices, where each rank-1 matrix is defined by distributional representation $\mathbf{v}_i$ and the weight difference between the weight of true label $\mathbf{w}_{y_i}$ and the "expected" weight $\sum_m \eta_{i,m}\mathbf{w}_m$.

One property of the dual variables is that $\mathbf{f}(\mathbf{v}_i;\mathbf{A})$ is a support vector only if the dual variable $\eta_{i,y_i} < 1$. Since the dual variables for each instance are guaranteed to sum to one, we have $\mathbf{w}_{y_i} - \sum_m \eta_{i,m}\mathbf{w}_m = 0$ if $\eta_{i,y_i} = 1$. In other words, the contribution from non support vectors to the projection matrix $\mathbf{A}$ is 0. Then, we can further simplify the updating equation as

$$\mathbf{A} = \frac{1}{\tau}\sum_{\mathbf{v}_i \in \text{SV}}(\mathbf{w}_{y_i} - \sum_m \eta_{i,m}\mathbf{w}_m)\mathbf{v}_i{}^\top \tag{10}$$

This is computationally advantageous since many instances are not support vectors, and it shows that the discriminatively-trained projection matrix only incorporates information from each instance to the extent that the correct classification receives low confidence.

Algorithm 1 Mini-batch learning algorithm

```
Input: Training set $\mathcal{D}$, Regularization parameters $\lambda$ and $\tau$, Number of iteration $T$, Initialization matrix $\mathbf{A}_0$, and Threshold $\varepsilon$
while $t = 1, \ldots, T$ do
    Randomly choose a subset of training samples $\mathcal{D}_t$ from $\mathcal{D}$
    Train SVM with $\mathbf{A}_{t-1}$ to obtain $\{\mathbf{w}_m^{(t)}\}$ and $\{\eta_{i,m}^{(t)}\}$
    Update $\mathbf{A}_t$ using Equation 11 with $\alpha_t = \frac{1}{t}$
    if $\frac{\|\mathbf{A}_t - \mathbf{A}_{t-1}\|_F}{\|\mathbf{A}_2 - \mathbf{A}_1\|_F} < \varepsilon$ then
        Return
    end if
end while
Re-train SVM with $\mathcal{D}$ and the final $\mathbf{A}$
Output: Projection matrix $\mathbf{A}$, SVM classifier with weights $\mathbf{w}$
```

3.2 Gradient-based Learning for A

Solving the quadratic programming defined by the dual form of the SVM is time-consuming, especially on a large-scale dataset. But if we focus on learning the projection matrix $\mathbf{A}$, we can speed up learning by sampling only a small proportion of the training data to compute an approximate optimum for $\{\mathbf{w}_{1:C}, \eta_{1:l,1:C}\}$, before each update of $\mathbf{A}$. This idea is similar to the mini-batch learning, which has been used in large-scale SVM problem (Nelakanti et al., 2013) and deep learning models (Le et al., 2011).

Specifically, in iteration t, the algorithm randomly chooses a subset of training samples $\mathcal{D}_t$ to train the model. We cannot make a closed-form update to $\mathbf{A}$ based on this small sample, but we can take an approximate gradient step,

$$\mathbf{A}_t = (1 - \alpha_t \tau)\mathbf{A}_{t-1} + \alpha_t \Big\{ \sum_{\mathbf{v}_i \in \mathrm{SV}(\mathcal{D}_t)} \Big(\mathbf{w}_{y_i}^{(t)} - \sum_m \eta_{i,m}^{(t)} \mathbf{w}_m^{(t)} \Big) \mathbf{v}_i{}^{\top} \Big\}, \quad (11)$$

where α_t is a learning rate. In iteration t, we choose $\alpha_t = \frac{1}{t}$. After convergence, we obtain the weights $\mathbf{w}$ by applying the SVM over the entire dataset, using the final $\mathbf{A}$. The algorithm is summarized in Algorithm 1 and more details about implementation will be clarified in Section 4. While minibatch learning requires more iterations, the SVM training is much faster in each batch, and the overall algorithm is several times faster than using the entire training set for each update.

4 Implementation

The learning algorithm is applied in a shift-reduce parser, where the training data consists of the (unique) list of shift and reduce operations required to produce the gold RST parses. On test data, we choose parsing operations in an online fashion — at each step, the parsing algorithm changes the status of the stack and the queue according the selected transition, then creates the next sample with the updated status.

4.1 Parameters and Initialization

There are three free parameters in our approach: the latent dimension K, and regularization parameters λ and τ. We consider the values $K \in \{30, 60, 90, 150\}$, $\lambda \in \{1, 10, 50, 100\}$ and $\tau \in \{1.0, 0.1, 0.01, 0.001\}$, and search over this space using a development set of thirty document randomly selected from within the RST Treebank training data. We initialize each element of $\mathbf{A}_0$ to a uniform random value in the range $[0, 1]$. For mini-batch learning, we fixed the batch size to be 500 training samples (shift-reduce operations) in each iteration.

4.2 Additional features

As described thus far, our model considers only the projected representation of each EDU in its parsing decisions. But prior work has shown that other, structural features can provide useful information (Joty et al., 2013). We therefore augment our classifier with a set of simple feature templates. These templates are applied to individual EDUs, as well as pairs of EDUs: (1) the two EDUs on top of the stack, and (2) the EDU on top of the stack and the EDU in front of the queue. The features are shown in Table 2. In computing these features, all tokens are downcased, and numerical features are not binned. The dependency structure and POS tags are obtained from MALTParser (Nivre et al., 2007).

5 Experiments

We evaluate DPLP on the RST Discourse Treebank (Carlson et al., 2001), comparing against state-of-the-art results. We also investigate the information encoded by the projection matrix.

5.1 Experimental Setup

Dataset The RST Discourse Treebank (RST-DT) consists of 385 documents, with 347 for train-

Feature	Examples
Words at beginning and end of the EDU	⟨BEGIN-WORD-STACK1 = *but*⟩ ⟨BEGIN-WORD-STACK1-QUEUE1 = *but, the*⟩
POS tag at beginning and end of the EDU	⟨BEGIN-TAG-STACK1 = CC⟩ ⟨BEGIN-TAG-STACK1-QUEUE1 = CC, DT⟩
Head word set from each EDU. The set includes words whose parent in the depenency graph is ROOT or is not within the EDU (Sagae, 2009).	⟨HEAD-WORDS-STACK2 = *working*⟩
Length of EDU in tokens	⟨LEN-STACK1-STACK2 = $\langle 7, 8\rangle$⟩
Distance between EDUs	⟨DIST-STACK1-QUEUE1 = 2⟩
Distance from the EDU to the beginning of the document	⟨DIST-FROM-START-QUEUE1 = 3⟩
Distance from the EDU to the end of the document	⟨DIST-FROM-END-STACK1 = 1⟩
Whether two EDUs are in the same sentence	⟨SAME-SENT-STACK1-QUEUE1 = True⟩

Table 2: Additional features for RST parsing

ing and 38 for testing in the standard split. As we focus on relational discourse parsing, we follow prior work (Feng and Hirst, 2012; Joty et al., 2013), and use gold EDU segmentations. The strongest automated RST segmentation methods currently attain 95% accuracy (Xuan Bach et al., 2012).

Preprocessing In the RST-DT, most nodes have exactly two children, one nucleus and one satellite. For non-binary relations, we use right-branching to binarize the tree structure. For multi-nuclear relations, we choose the left EDU as "head" EDU. The vocabulary $\mathcal{V}$ includes all unigrams after down-casing. No other preprocessing is performed. In total, there are 16250 unique unigrams in $\mathcal{V}$.

Fixed projection matrix baselines Instead of learning from data, a simple way to obtain a projection matrix is to use matrix factorization. Recent work has demonstrated the effectiveness of non-negative matrix factorization (NMF) for measuring distributional similarity (Dinu and Lapata, 2010; Van de Cruys and Apidianaki, 2011). We can construct $\mathbf{B}_{nmf}$ in the *concatenation form* of the projection matrix by applying NMF to the EDU-feature matrix, $\mathbf{M} \approx \mathbf{WH}$. As a result, $\mathbf{W}$ describes each EDU with a K-dimensional vector, and $\mathbf{H}$ describes each word with a K-dimensional vector. We can then construct $\mathbf{B}_{nmf}$ by taking the pseudo-inverse of $\mathbf{H}$, which then projects from word-count vectors into the latent space.

Another way to construct $\mathbf{B}$ is to use neural word embeddings (Collobert and Weston, 2008). In this case, we can view the product $\mathbf{Bv}$ as a composition of the word embeddings, using the simple *additive* composition model proposed by Mitchell and Lapata (2010). We used the word embeddings from Collobert and Weston (2008) with dimension $\{25, 50, 100\}$. Grid search over heldout training data was used to select the optimum latent dimension for both the NMF and word embedding baselines. Note that the size K of the resulting projection matrix is three times the size of the embedding (or NMF representation) due to the concatenate construction.

We also consider the special case where $\mathbf{A} = \mathbf{I}$.

Competitive systems We compare our approach with HILDA (Hernault et al., 2010) and TSP (Joty et al., 2013). Joty et al. (2013) proposed two different approaches to combine sentence-level parsing models: *sliding windows* (TSP SW) and *1 sentence-1 subtree* (TSP 1-1). In the comparison, we report the results of both approaches. All results are based on the same gold standard EDU segmentation. We cannot compare with the results of Feng and Hirst (2012), because they do not evaluate on the overall discourse *structure*, but rather treat each relation as an individual classification problem.

Metrics To evaluate the parsing performance, we use the three standard ways to measure the performance: unlabeled (i.e., hierarchical spans) and labeled (i.e., nuclearity and relation) F-score, as defined by Black et al. (1991). The application of this approach to RST parsing is described by Marcu (2000b).[3] To compare with previous works on RST-DT, we use the 18 coarse-grained relations defined in (Carlson et al., 2001).

[3] We implemented the evaluation metrics by ourselves. Together with the DPLP system, all codes are published on `https://github.com/jiyfeng/DPLP`

Method	Matrix Form	+Features	K	Span	Nuclearity	Relation
Prior work						
1. HILDA (Hernault *et al.*, 2010)				**83.0**	68.4	54.8
2. TSP 1-1 (Joty *et al.*, 2013)				82.47	68.43	55.73
3. TSP SW (Joty *et al.*, 2013)				82.74	68.40	55.71
Our work						
4. Basic features	$\mathbf{A} = \mathbf{0}$	Yes		79.43	67.98	52.96
5. Word embeddings	Concatenation	No	75	75.28	67.14	53.79
6. NMF	Concatenation	No	150	78.57	67.66	54.80
7. Bag-of-words	$\mathbf{A} = \mathbf{I}$	Yes		79.85	69.01	60.21
8. DPLP	Concatenation	No	60	80.91	69.39	58.96
9. DPLP	Difference	No	60	80.47	68.61	58.27
10. DPLP	Concatenation	Yes	60	82.08	**71.13**	**61.63**
11. DPLP	General	Yes	30	81.60	**70.95**	**61.75**
Human annotation				88.70	77.72	65.75

Table 3: Parsing results of different models on the RST-DT test set. The results of TSP and HILDA are reprinted from prior work (Joty et al., 2013; Hernault et al., 2010).

5.2 Experimental Results

Table 3 presents RST parsing results for DPLP and some alternative systems. All versions of DPLP outperform the prior state-of-the-art on nuclearity and relation detection. This includes relatively simple systems whose features are simply a projection of the word count vectors for each EDU (lines 7 and 8). The addition of the features from Table 2 improves performance further, leading to absolute F-score improvement of around 2.5% in nuclearity and 6% in relation prediction (lines 9 and 10).

On span detection, DPLP performs slightly worse than the prior state-of-the-art. These systems employ richer syntactic and contextual features, which might be especially helpful for span identification. As shown by line 4 of the results table, the basic features from Table 2 provide most of the predictive power for spans; however, these features are inadequate at the more semantically-oriented tasks of nuclearity and relation prediction, which benefit substantially from the projected features. Since correctly identifying spans is a precondition for nuclearity and relation prediction, we might obtain still better results by combining features from HILDA and TSP with the representation learning approach described here.

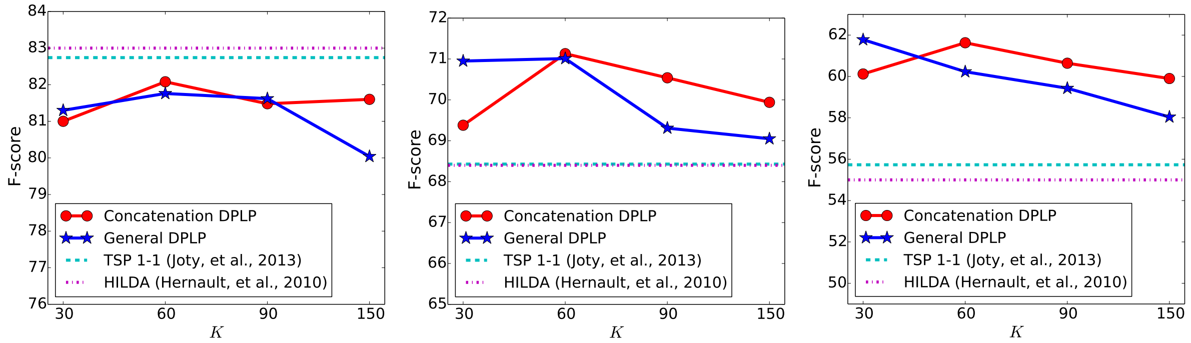
Lines 5 and 6 show that discriminative learning of the projection matrix is crucial, as fixed projections obtained from NMF or neural word embeddings perform substantially worse. Line 7 shows that the original bag-of-words representation together with basic features could give us some benefit on discourse parsing, but still not as good as results from DPLP. From lines 8 and 9, we see that the concatenation construction is superior to the difference construction, but the comparison between lines 10 and 11 is inconclusive on the merits of the general form of $\mathbf{A}$. This suggests that using the projection matrix to model interrelationships between EDUs does not substantially improve performance, and the simpler concatenation construction may be preferred.

Figure 3 shows how performance changes for different latent dimensions K. At each value of K, we employ grid search over a development set to identify the optimal regularizers λ and τ. For the concatenation construction, performance is not overly sensitive to K. For the general form of $\mathbf{A}$, performance decreases with large K. Recall from Section 2.3 that this construction has nine times as many parameters as the concatenation form; with large values of K, it is likely to overfit.

5.3 Analysis of Projection Matrix

Why does projection of the surface features improve discourse parsing? To answer this question, we examine what information the projection matrix is learning to encoded. We take the projection matrix from the concatenation construction and $K = 60$ as an example for case study. Recalling the definition in equation 4, the projection matrix $\mathbf{A}$ will be composed of three identical submatrices $\mathbf{B} \in \mathbb{R}^{20 \times V}$. The columns of the $\mathbf{B}$ matrix can be viewed as 20-dimensional descriptors of the words in the vocabulary.

For the purpose of visualization, we further reduce the dimension of latent representation from $K = 20$ to 2 dimensions using t-SNE (van der Maaten and Hinton, 2008). One further simpli-



(a) Span

(b) Nuclearity

(c) Relation

Figure 3: The performance of our parser over different latent dimension K. Results for DPLP include the additional features from Table 3

fication for visualization is we consider only the top 1000 frequent unigrams in the RST-DT training set. For comparison, we also apply t-SNE to the projection matrix $\mathbf{B}_{nmf}$ recovered from non-negative matrix factorization.

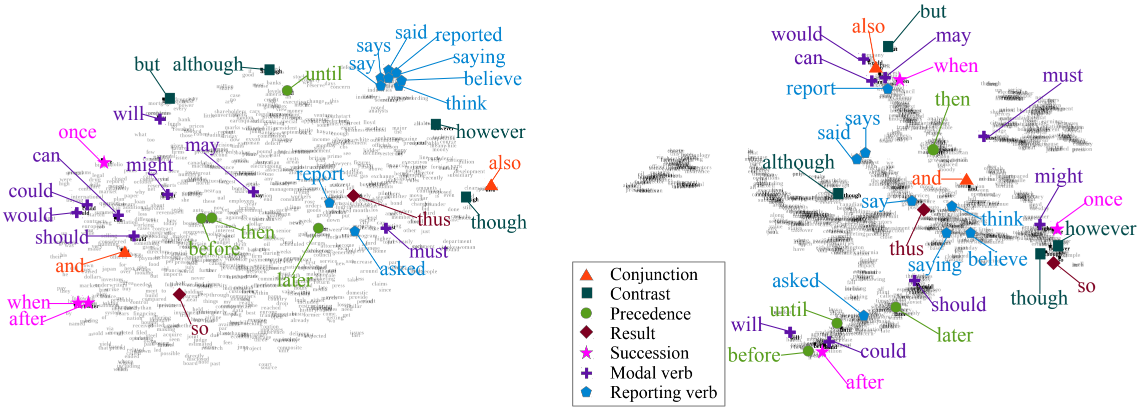
Figure 4 highlights words that are related to discourse analysis. Among the top 1000 words, we highlight the words from 5 major discourse connective categories provided in Appendix B of the PDTB annotation manual (Prasad et al., 2008): CONJUNCTION, CONTRAST, PRECEDENCE, RESULT, and SUCCESSION. In addition, we also highlighted two verb categories from the top 1000 words: modal verbs and reporting verbs, with their inflections (Krestel et al., 2008).

From the figure, it is clear DPLP has learned a projection matrix that successfully groups several major discourse-related word classes: particularly modal and reporting verbs; it has also grouped succession and precedence connectives with some success. In contrast, while NMF does obtain compact clusters of words, these clusters appear to be completely unrelated to discourse function of the words that they include. This demonstrates the value of using discriminative training to obtain the transformed representation of the discourse units.

6 Related Work

Early work on document-level discourse parsing applied hand-crafted rules and heuristics to build trees in the framework of Rhetorical Structure Theory (Sumita et al., 1992; Corston-Oliver, 1998; Marcu, 2000a). An early data-driven approach was offered by Schilder (2002), who used distributional techniques to rate the topicality of each discourse unit, and then chose among underspecified discourse structures by placing more topical sentences near the root. Learning-based approaches were first applied to identify within-sentence discourse relations (Soricut and Marcu, 2003), and only later to cross-sentence relations at the document level (Baldridge and Lascarides, 2005). Of particular relevance to our inference technique are incremental discourse parsing approaches, such as shift-reduce (Sagae, 2009) and A* (Muller et al., 2012). Prior learning-based work has largely focused on lexical, syntactic, and structural features, but the close relationship between discourse structure and semantics (Forbes-Riley et al., 2006) suggests that shallow feature sets may struggle to capture the long tail of alternative lexicalizations that can be used to realize discourse relations (Prasad et al., 2010; Marcu and Echihabi, 2002). Only Subba and Di Eugenio (2009) incorporate rich compositional semantics into discourse parsing, but due to the ambiguity of their semantic parser, they must manually select the correct semantic parse from a forest of possiblities.

Recent work has succeeded in pushing the state-of-the-art in RST parsing by innovating on several fronts. Feng and Hirst (2012) explore rich linguistic linguistic features, including lexical semantics and discourse production rules suggested by Lin et al. (2009) in the context of the Penn Discourse Treebank (Prasad et al., 2008). Muller et al. (2012) show that A* decoding can outperform both greedy and graph-based decoding algorithms. Joty et al. (2013) achieve the best prior results on RST relation detection by (i) jointly performing relation detection and classification, (ii) performing bottom-up rather than greedy decoding, and (iii) distinguishing between intra-sentence and inter-sentence relations. Our approach is largely orthogonal to this prior work: we focus on trans-



(a) Latent representation of words from projection learning with $K = 20$.

(b) Latent representation of words from non-negative matrix factorization with $K = 20$.

Figure 4: t-SNE Visualization on latent representations of words.

forming the lexical representation of discourse units into a latent space to facilitate learning. As shown in Figure 4(a), this projection succeeds at grouping words with similar discourse functions. We might expect to obtain further improvements by augmenting this representation learning approach with rich syntactic features (particularly for span identification), more accurate decoding, and special treatment of intra-sentence relations; this is a direction for future research.

Discriminative learning of latent features for discourse processing can be viewed as a form of *representation learning* (Bengio et al., 2013). Also called Deep Learning, such approaches have recently been applied in a number of NLP tasks (Collobert et al., 2011; Socher et al., 2012). Of particular relevance are applications to the detection of semantic or discourse relations, such as paraphrase, by comparing sentences in an induced latent space (Socher et al., 2011; Guo and Diab, 2012; Ji and Eisenstein, 2013). In this work, we show how discourse structure annotations can function as a supervision signal to discriminatively learn a transformation from lexical features to a latent space that is well-suited for discourse parsing. Unlike much of the prior work on representation learning, we induce a simple linear transformation. Extension of our approach by incorporating a non-linear activation function is a natural topic for future research.

7 Conclusion

We have presented a framework to perform discourse parsing while jointly learning to project to a low-dimensional representation of the discourse units. Using the vector-space representation of EDUs, our shift-reduce parsing system substantially outperforms existing systems on nuclearity detection and discourse relation identification. By adding some additional surface features, we obtain further improvements. The low dimensional representation also captures basic intuitions about discourse connectives and verbs, as shown in Figure 4(a).

Deep learning approaches typically apply a non-linear transformation such as the sigmoid function (Bengio et al., 2013). We have conducted a few unsuccessful experiments with the "hard tanh" function proposed by Collobert and Weston (2008), but a more complete exploration of non-linear transformations must wait for future work. Another direction would be more sophisticated composition of the surface features within each elementary discourse unit, such as the hierarchical convolutional neural network (Kalchbrenner and Blunsom, 2013) or the recursive tensor network (Socher et al., 2013). It seems likely that a better accounting for syntax could improve the latent representations that our method induces.

Acknowledgments

We thank the reviewers for their helpful feedback, particularly for the connection to multitask learning. We also want to thank Kenji Sagae and Vanessa Wei Feng for the helpful discussion via email communication. This research was supported by Google Faculty Research Awards to the second author.

References

Jason Baldridge and Alex Lascarides. 2005. Probabilistic head-driven parsing for discourse structure. In *Proceedings of the Ninth Conference on Computational Natural Language Learning*, pages 96–103.

Yoshua Bengio, Aaron Courville, and Pascal Vincent. 2013. Representation Learning: A Review and New Perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828.

Ezra Black, Steve Abney, Dan Flickinger, Claudia Gdaniec, Ralph Grishman, Phil Harrison, Don Hindle, Robert Ingria, Fred Jelinek, Judith Klavans, Mark Liberman, Mitchell Marcus, Salim Roukos, Beatrice Santorini, and Tomek Strzalkowski. 1991. A Procedure for Quantitatively Comparing the Syntactic Coverage of English Grammars. In *Speech and Natural Language: Proceedings of a Workshop Held at Pacific Grove, California, February 19-22, 1991*, pages 306–311.

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

Lynn Carlson, Daniel Marcu, and Mary Ellen Okurowski. 2001. Building a Discourse-tagged Corpus in the Framework of Rhetorical Structure Theory. In *Proceedings of Second SIGdial Workshop on Discourse and Dialogue*.

Michael Collins and Brian Roark. 2004. Incremental parsing with the perceptron algorithm. In *Proceedings of ACL*, page 111. Association for Computational Linguistics.

R. Collobert and J. Weston. 2008. A Unified Architecture for Natural Language Processing: Deep Neural Networks with Multitask Learning. In *ICML*.

R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa. 2011. Natural Language Processing (Almost) from Scratch. *Journal of Machine Learning Research*, 12:2493–2537.

Simon Corston-Oliver. 1998. Beyond string matching and cue phrases: Improving efficiency and coverage in discourse analysis. In *The AAAI Spring Symposium on Intelligent Text Summarization*, pages 9–15.

Koby Crammer and Yoram Singer. 2001. On the Algorithmic Implementation of Multiclass Kernel-based Vector Machines. *Journal of Machine Learning Research*, 2:265–292.

Georgiana Dinu and Mirella Lapata. 2010. Measuring Distributional Similarity in Context. In *EMNLP*, pages 1162–1172.

Vanessa Wei Feng and Graeme Hirst. 2012. Text-level Discourse Parsing with Rich Linguistic Features. In *Proceedings of ACL*.

David Ferrucci, Eric Brown, Jennifer Chu-Carroll, James Fan, David Gondek, Aditya A Kalyanpur, Adam Lally, J William Murdock, Eric Nyberg, John Prager, et al. 2010. Building Watson: An overview of the DeepQA project. *AI magazine*, 31(3):59–79.

Katherine Forbes-Riley, Bonnie Webber, and Aravind Joshi. 2006. Computing discourse semantics: The predicate-argument semantics of discourse connectives in D-LTAG. *Journal of Semantics*, 23(1):55–106.

Weiwei Guo and Mona Diab. 2012. Modeling Sentences in the Latent Space. In *Proceedings of ACL*, pages 864–872, Jeju Island, Korea, July. Association for Computational Linguistics.

Hugo Hernault, Helmut Prendinger, David A. duVerle, and Mitsuru Ishizuka. 2010. HILDA: A Discourse Parser Using Support Vector Machine Classification. *Dialogue and Discourse*, 1(3):1–33.

Yangfeng Ji and Jacob Eisenstein. 2013. Discriminative Improvements to Distributional Sentence Similarity. In *EMNLP*, pages 891–896, Seattle, Washington, USA, October. Association for Computational Linguistics.

Shafiq Joty, Giuseppe Carenini, Raymond Ng, and Yashar Mehdad. 2013. Combining Intra- and Multi-sentential Rhetorical Parsing for Document-level Discourse Analysis. In *Proceedings of ACL*.

Nal Kalchbrenner and Phil Blunsom. 2013. Recurrent convolutional neural networks for discourse compositionality. In *Proceedings of the Workshop on Continuous Vector Space Models and their Compositionality*, pages 119–126, Sofia, Bulgaria, August. Association for Computational Linguistics.

Terry Koo, Xavier Carreras, and Michael Collins. 2008. Simple Semi-supervised Dependency Parsing. In *Proceedings of ACL-HLT*, pages 595–603, Columbus, Ohio, June. Association for Computational Linguistics.

Ralf Krestel, Sabine Bergler, and René Witte. 2008. Minding the Source: Automatic Tagging of Reported Speech in Newspaper Articles. In *LREC*, Marrakech, Morocco, May. European Language Resources Association (ELRA).

Quoc V. Le, Jiquan Ngiam, Adam Coates, Abhik Lahiri, Bobby Prochnow, and Andrew Y. Ng. 2011. On Optimization Methods for Deep Learning. In *ICML*.

Ziheng Lin, Min-Yen Kan, and Hwee Tou Ng. 2009. Recognizing Implicit Discourse Relations in the Penn Discourse Treebank. In *EMNLP*.

Annie Louis, Aravind Joshi, and Ani Nenkova. 2010. Discourse indicators for content selection in summarization. In *Proceedings of the 11th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 147–156. Association for Computational Linguistics.

Daniel Marcu and Abdessamad Echihabi. 2002. An Unsupervised Approach to Recognizing Discourse Relations. In *Proceedings of ACL*, pages 368–375, Philadelphia, Pennsylvania, USA, July. Association for Computational Linguistics.

Daniel Marcu. 1996. Building Up Rhetorical Structure Trees. In *Proceedings of AAAI*.

Daniel Marcu. 1999. A Decision-Based Approach to Rhetorical Parsing. In *Proceedings of ACL*, pages 365–372, College Park, Maryland, USA, June. Association for Computational Linguistics.

Daniel Marcu. 2000a. The Rhetorical Parsing of Unrestricted Texts: A Surface-based Approach. *Computational Linguistics*, 26:395–448.

Daniel Marcu. 2000b. *The Theory and Practice of Discourse Parsing and Summarization*. MIT Press.

Scott Miller, Jethran Guinness, and Alex Zamanian. 2004. Name Tagging with Word Clusters and Discriminative Training. In Daniel Marcu Susan Dumais and Salim Roukos, editors, *HLT-NAACL*, pages 337–342, Boston, Massachusetts, USA, May 2 - May 7. Association for Computational Linguistics.

Eleni Miltsakaki and Karen Kukich. 2004. Evaluation of text coherence for electronic essay scoring systems. *Natural Language Engineering*, 10(1):25–55.

Jeff Mitchell and Mirella Lapata. 2010. Composition in distributional models of semantics. *Cognitive Science*, 34(8):1388–1429.

Philippe Muller, Stergos Afantenos, Pascal Denis, and Nicholas Asher. 2012. Constrained Decoding for Text-Level Discourse Parsing. In *Coling*, pages 1883–1900, Mumbai, India, December. The COLING 2012 Organizing Committee.

Anil Kumar Nelakanti, Cedric Archambeau, Julien Mairal, Francis Bach, and Guillaume Bouchard. 2013. Structured Penalties for Log-Linear Language Models. In *EMNLP*, pages 233–243, Seattle, Washington, USA, October. Association for Computational Linguistics.

Joakim Nivre, Johan Hall, Jens Nilsson, Atanas Chanev, Gülsen Eryigit, Sandra Kübler, Svetoslav Marinov, and Erwin Marsi. 2007. MaltParser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering*, 13(2):95–135.

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

Rashmi Prasad, Aravind Joshi, and Bonnie Webber. 2010. Realization of discourse relations by other means: alternative lexicalizations. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pages 1023–1031. Association for Computational Linguistics.

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 (IWPT)*, pages 81–84, Paris, France, October. Association for Computational Linguistics.

Frank Schilder. 2002. Robust discourse parsing via discourse markers, topicality and position. *Natural Language Engineering*, 8(3):235–255.

Richard Socher, Eric H. Huang, Jeffrey Pennington, Andrew Y. Ng, and Christopher D. Manning. 2011. Dynamic Pooling and Unfolding Recursive Autoencoders for Paraphrase Detection. In *NIPS*.

Richard Socher, Brody Huval, Christopher D. Manning, and Andrew Y. Ng. 2012. Semantic Compositionality Through Recursive Matrix-Vector Spaces. In *EMNLP*.

Richard Socher, Alex Perelygin, Jean Y Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Swapna Somasundaran, Galileo Namata, Janyce Wiebe, and Lise Getoor. 2009. Supervised and unsupervised methods in employing discourse relations for improving opinion polarity classification. In *Proceedings of EMNLP*.

Radu Soricut and Daniel Marcu. 2003. Sentence Level Discourse Parsing using Syntactic and Lexical Information. In *NAACL*.

Rajen Subba and Barbara Di Eugenio. 2009. An effective Discourse Parser that uses Rich Linguistic Information. In *NAACL-HLT*, pages 566–574, Boulder, Colorado, June. Association for Computational Linguistics.

K. Sumita, K. Ono, T. Chino, T. Ukita, and S. Amano. 1992. A discourse structure analyzer for Japanese text. In *Proceedings International Conference on Fifth Generation Computer Systems*, pages 1133–1140.

Maite Taboada and William C Mann. 2006. Applications of rhetorical structure theory. *Discourse studies*, 8(4):567–588.

Benjamin Taskar, Carlos Guestrin, and Daphne Koller. 2003. Max-margin markov networks. In *NIPS*.

Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. Word Representation: A Simple and General Method for Semi-Supervised Learning. In *Proceedings of ACL*, pages 384–394.

Tim Van de Cruys and Marianna Apidianaki. 2011. Latent Semantic Word Sense Induction and Disambiguation. In *Proceedings of ACL*, pages 1476–1485, Portland, Oregon, USA, June. Association for Computational Linguistics.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing Data using t-SNE. *Journal of Machine Learning Research*, 9:2759–2605, November.

Kimberly Voll and Maite Taboada. 2007. Not all words are created equal: Extracting semantic orientation as a function of adjective relevance. In *Proceedings of Australian Conference on Artificial Intelligence*.

Ngo Xuan Bach, Nguyen Le Minh, and Akira Shimazu. 2012. A Reranking Model for Discourse Segmentation using Subtree Features. In *Proceedings of the 13th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 160–168.

Chun-Nam John Yu and Thorsten Joachims. 2009. Learning structural SVMs with latent variables. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 1169–1176. ACM.