SEMANTIC REPRESENTATION LEARNING FOR DISCOURSE PROCESSING

A Thesis Proposal
Presented to
The Academic Faculty

by

Yangfeng Ji

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Interactive Computing

Georgia Institute of Technology
August 2016

SEMANTIC REPRESENTATION LEARNING FOR DISCOURSE PROCESSING

Approved by:

Professor Jacob Eisenstein, Advisor
School of Interactive Computing
Georgia Institute of Technology

Professor Noah Smith
Department of Computer Science & Engineering
University of Washington

Professor Mark Riedl
School of Interactive Computing
Georgia Institute of Technology

Professor Chris Dyer
School of Computer Science
Carnegie Mellon University

Professor Byron Boots
School of Interactive Computing
Georgia Institute of Technology

Date Approved: April 27, 2016

ACKNOWLEDGEMENTS

First, I would like to thank Jacob Eisenstein for being a fantastic and insightful advisor. He gave an opportunity to continue my dream of being a computer scientist, helped me to find my research interests and my thesis topic. Working with him is a joyful and fruitful experience. He basically teached everything about research, even including how to debug my MATLAB code (unfortunately, only in my first semester).

Thank my thesis committee members: Byron Boots, Chris Dyer, Mark Riedl and Noah Smith for their feedback and suggestions on my work, and for helping polishing my dissertation. Special thank to Noah for reading my dissertation word by word. I feel more confident about my writing after revising with his comments.

The School of Interactive Computing at Georgia Tech has been a great place to study and work for the last 4 years. As a student transfered from another school, I am lucky to have lots of friends for consistent support on my work. I want to thank the student members and alumni in the Computational Linguistics Group (Yi Yang, Umashanthi Pavalanathan, Sandeep Soni, Ian Stewart, Naman Goyal, Akanksha, Rahul Goel, Yijie Wang, Ana Smith, Vinodh Krishnan, Gongbo Zhang and Caglar Tirkaz) for their consistent support of my research. Also I would like to thank my friends from not only Georgia Tech but some other places (Boyang Li, Hongliang Li, Lei Liu, Lu Lu, Yun Wei, Liang Wu, Bo Xie, Cong Xiong, Richard Rutledge, Yi Yang, Zhaoming Yin, Hong Yu, Xueyun Zhu) for their help in the past several years.

I must also thank the entire machine translation team in the JSALT workshop 2015 at University of Washington. It gave me a chance to experience a different style of research and also got to know lots of excellent researchers in NLP and machine learning. While at Seattle, I worked with many terrific collaborators and friends:

Trevor Cohn, Chris Dyer, Kevin Duh, Reza Haffari, Lingpeng Kong, Yi Luan and (of course) Jacob Eisenstein. Some of them have became my long-term collaborators, and I am still enjoying work with them. The visiting to University of Washington also made me realize that it has excellent community on both Linguistics and NLP. So I decided to find myself a position there — thank Noah for providing me a job.

I spent two summers at Microsoft Research as research interns. MSR is not just a place where I can get endless free StarBucks coffee, but also a place where I can explore some new research interests, like conversational modeling. I want to thank my mentors Dilek Hakkani-Tur (summer 2013) and Michel Galley (summer 2014), and my collaborators at Speech group and NLP group: Asli Celikyilmaz, Larry Heck, Gokhan Tur, Alessandro Sordoni, Michel Galley, Michael Auli, Chris Brockett, Meg Mitchell, Jianfeng Gao and Bill Dolan.

Finally, my deep thanks for the love and support from my family. To my dad and mom for their unconditional love on whatever I want to do. To my grandfather — I miss you! To my lovely wife Linejie for always being with me and going through all the troubles together. To my little angel Sophie!

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

Discourse processing is to identify coherent relations, such as contrast and causal relation, from well-organized texts. The outcomes from discourse processing can benefit both research and applications in natural language processing, such as recognizing the major opinion from a product review, or evaluating the coherence of student writings. Identifying discourse relations from texts is an essential task of discourse processing. Relation identification requires intensive semantic understanding of texts, especially when no word (e.g., *but*) can signal the relations. Most prior work relies on sparse representation constructed from surface-form features (including, word pairs, POS tags, etc.), which fails to encode enough semantic information. As an alternative, I propose to use distributed representations of texts, which are dense vectors and flexible enough to share information efficiently.

The goal of my work is to develop new models with representation learning for discourse processing. Specifically, I present a unified framework in this thesis to be able to learn both distributed representation and discourse models jointly. The joint training not only learns the discourse models, but also helps to shape the distributed representation for the discourse models. Such that, the learned representation could encode necessary semantic information to facilitate the processing tasks. The evaluation shows that our systems outperform prior work with only surface-form representations. In this thesis, I also discuss the possibility of extending the representation learning framework into some other problems in discourse processing. The problems studied include (1) How to use representation learning to build a discourse model with only distant supervision? The investigation of this problem will help to reduce

the dependency of discourse processing on the annotated data; (2) How to combine discourse processing with other NLP tasks, such as language modeling? The exploration of this problem is expected to show the value of discourse information, and draw more attention to the research of discourse processing. As the end of this thesis, it also demonstrates the benefit of using discourse information for document-level machine translation and sentiment analysis.

CHAPTER I

INTRODUCTION

A natural language text does not normally consist of isolated sentences, but of sentences in a coherent order. We call a coherent text as a **discourse**, in order to emphasize that the information conveyed by a discourse is usually larger than the sum of information from individual sentences. Typically, the coherence of discourse is characterized by several intrinsic features, such as the position of sentences, the order of sentences, the connection of adjacent sentences and the context from surrounding text [205]. These intrinsic features together imply a simple fact that "discourse has structure" [71].

In natural language processing (NLP), we need discourse structure to help computers capture the structural organization of a text, understand the main content of a document, and generate a new coherent text. For example, the information embedded in discourse structure can help sentiment analysis to get accurate results on product reviews (Figure 1(a)), or it can make translated texts more fluent (Figure 1(b)). Some other applications of discourse information include document coherence evaluation [78, 143, 21], text generation [75], document summarization [121] and question answering [51, 77][1].

Discourse processing is the language technology to extract discourse structures from documents automatically. The research in discourse processing covers segmenting documents based on topic structures (*topic segmentation*), identifying anaphoric structures of documents (*anaphora/coreference resolution*), and analyzing coherence structures of documents (*discourse parsing*) [185]. In this thesis, I mainly focus on

[1]For more applications about discourse information, refer to [205, 189] and chapter 6.

discourse parsing, since this research topic is more close to semantic understand of texts than the others.

There was only limited published work compared to other research areas in NLP several years ago. The research progress on discourse processing has been dragged down for more than a decade because of the semantic representation issue. To identify the discourse structure of a text, especially the discourse relations (e.g.; CAUSE, CONTRAST or TEMPORAL), requires a discourse processing system to capture the semantic information between text units. However, the representation methods from previous work are not flexible enough to encode sufficient semantic information. Consider the example 1, in order to identify the discourse relation from the two sentences, a human reader can easily understand that

(i) "*She*" refers to "*Tina*";

(ii) "*Burger*" is a food which can relieve hunger, and

(iii) the reason Bob gave the burger to Tina is that "*Tina*" was hungry.

Therefore, the relation between these two sentences is: sentence (2) provides a CAUSE for the action described in sentence (1).

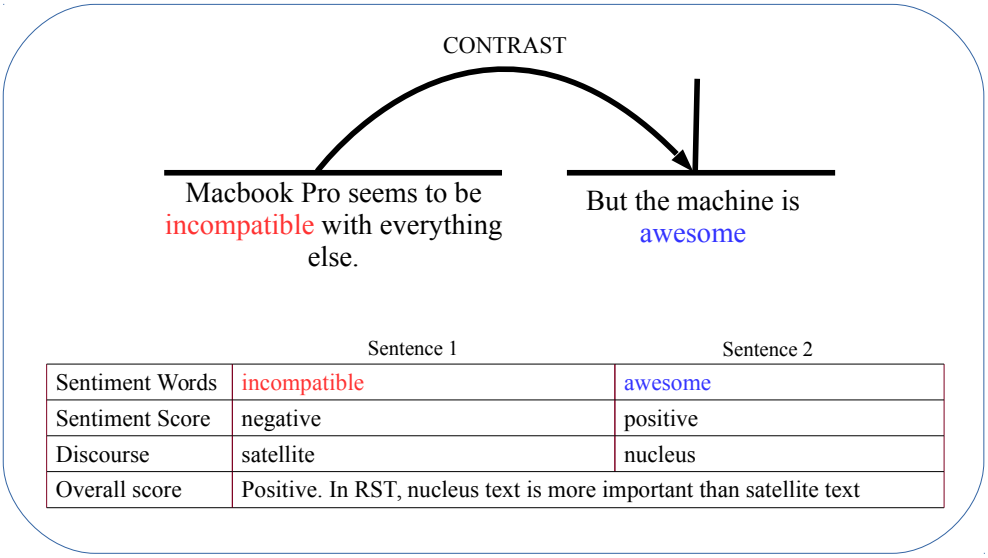
Example 1.

What is the discourse relation between two sentences:

(1) Bob gave Tina the burger.

(2) She was hungry.

Theoretically, human reasoning can be simulated by a computer with logical representations of sentences and an inference engine with logical power, as shown in Figure 2(a). Prior work [70] argues that discourse relations can be inferred based on the logical representations of sentences and a large set of inference rules with some



	Sentence 1	Sentence 2
Sentiment Words	incompatible	awesome
Sentiment Score	negative	positive
Discourse	satellite	nucleus
Overall score	Positive. In RST, nucleus text is more important than satellite text	

(a) Discourse structure could identify which part of a text is more important then others with respect to the content. In this example, the second sentence is more central than the first sentence. Therefore, the overall sentiment score is positive, instead of neutral.

Original text:
亚马逊将会大大降低它的运输成本，如果可以使用无人机送货。

Google translation:
Amazon will significantly reduce its transportation costs, if you can use the UAV delivery

Discourse-ware translation:

Discourse information:
(1) First clause – CONDITION – Second clause
(2) A subject is missing from the second clause

Discourse-aware decoding:
(1) The second clause has the same subject as the first

Final translation:
Amazon will significantly reduce its transportation costs, if it can use the UAV delivery

(b) Discourse information could be used to eliminate translation errors with respect to context. In this example, the second clause as the condition of the first, it should also have the same subject as the first clause.

Figure 1: Two examples about using discourse information in NLP applications. More detail about discourse structure will be discussed in chapter 2.

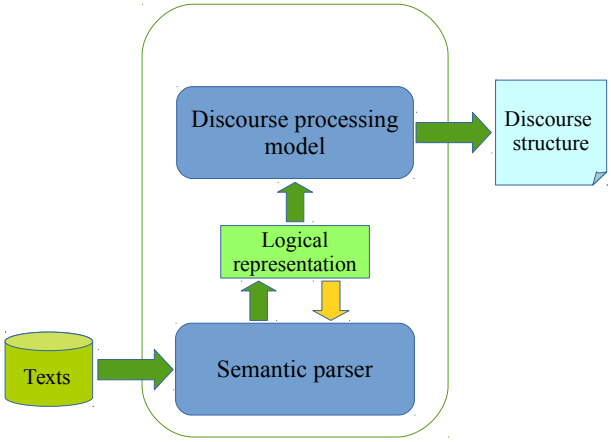
world knowledge. However, to implement this system, we will need a high-precision semantic parser for getting logical representations [172], and a robust knowledge base for constructing enough inference rules. Semantic parsing [146] and reasoning with knowledge base [61] are two fruitful research areas. Combining them together for automatic discourse processing will be an attractive research topic in the future, but it is not feasible for now.

To get around the difficulty of building logical representations, researchers propose an alternative way to represent texts for discourse processing, called *surface-form representation* [185]. This representation utilizes the surface features including words, phrases or other components extracted from a text and its syntactic parses. For instance, simple features for the discourse in Example 1 are the word pairs from the sentences, like $\langle the, was\rangle$, $\langle burger, hungry\rangle$ and $\langle Tina, She\rangle$, etc. These features are easily converted into numeric vectors with a manually-designed feature vocabulary [86]. Then, discourse processing is performed by some machine learning models with numeric operations. The advantage is that these machine learning models are learned from annotated corpora, without any extra world knowledge[2]. This representation methodology has facilitated discourse processing for many years. Figure 2(b) illustrates the typical framework of a discourse processing system with machine learning models and surface-form representation.

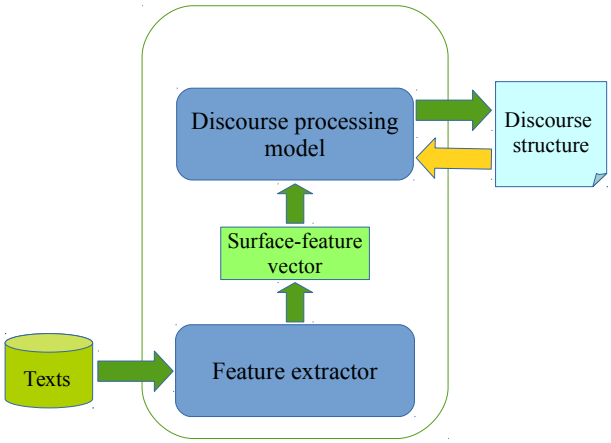
1.1 The Problem of Surface-form Representation

However, representation with surface features has its fundamental limitations from the perspective of machine learning and linguistics. Consider the example 1. If we replace the word "*burger*" with "*donut*", or "*hungry*" with "*starving*", the relation between these two sentences is still the same. An ideal representation on semantics

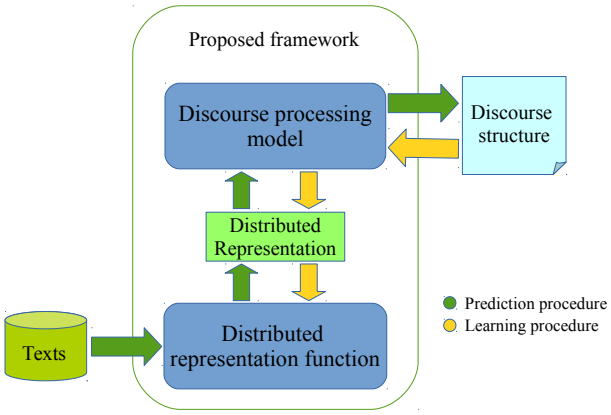
[2]Though it should be noted that annotating discourse information on texts definitely needs the annotators' world knowledge and even professional training.



(a) Discourse processing with logical representations from semantic parser



(b) Discourse processing with surface features



(c) The proposed framework for distributed semantic representation and discourse processing.

Figure 2: Three frameworks on discourse processing

should reflect the semantic similarity between these words. Unfortunately, surface-form representation is limited by the amount of semantic information it can represent.

To make it clear, consider a simple surface-form representation in NLP called the bag-of-words representation. To build a bag-of-words representation, we first need a vocabulary constructed on a training corpus. Then, every text is represented as a sparse numeric vector, which has the same size of the vocabulary. Figure 3 shows the surface-form representation of some example words. Every word in this representation has a single non-zero element, and all the rest elements are zeros. An immediate problem of this representation is the amount of semantic information. For example, if we use a numeric distance between two word vectors as the metric of the semantic (dis-)similarity, then we have

$$\text{dist}(\boldsymbol{w}_{\texttt{burger}}, \boldsymbol{w}_{\texttt{donut}}) = \text{dist}(\boldsymbol{w}_{\texttt{burger}}, \boldsymbol{w}_{\texttt{gave}}) \tag{1}$$

where $\text{dist}(\cdot, \cdot)$ can be any distance function defined in mathematics, e.g., ℓ_1 norm or ℓ_2 norm. On the other hand, we all understand that the meaning of word "*donut*" should be close to "*burger*" then "*gave*". The example in Equation 1 demonstrates the limitation of a surface-form representation from the linguistic perspective.

Another critical limitation of a surface-form representation is that the information learned on a linguistic term cannot be transferred to a similar term. Recall that the discourse processing models combined with surface-form representation (as shown in 2(b)) are essentially some machine learning models. A better machine learning model is expected to be able to generalize what it learned from the training data onto some unobserved test data, which is called the generalization power [15]. However, generalization with surface-form representation is not straightforward. Consider $\boldsymbol{w}_{\texttt{burger}}$ and $\boldsymbol{w}_{\texttt{donut}}$, because they have different non-zero elements, the information learned about "*burger*" cannot be transferred to "*donut*". To obtain a model that can handle the word "*donut*", we need to make sure that this word is also contained in the training corpus explicitly. In practice, it requires us to collect a huge amount of training data

Vocabulary:	$\cdots$	donut	gave	hungry	starving	burger	$\cdots$
$\boldsymbol{w}_{\texttt{burger}}$	$[\cdots$	0	0	0	0	1	$\cdots]$
$\boldsymbol{w}_{\texttt{donut}}$	$[\cdots$	1	0	0	0	0	$\cdots]$
$\boldsymbol{w}_{\texttt{gave}}$	$[\cdots$	0	1	0	0	0	$\cdots]$

Figure 3: A surface-form representation of some example words. Every word is represented as a sparse numeric vector based the vocabulary.

in order to cover every possible cases, which is usually infeasible.

Besides these two major issues, there are some other limitations of surface-form representation. For example, surface features depend heavily on the design of a feature extractor, as shown in Figure 2(b). In practice, a manually-designed feature extractor is not easily adaptable across different tasks. Surface features extracted from one task may not be representative of another.

In the above example, I only discuss with words to demonstrate the problem of surface-form representation. But the conclusions are also applied to the surface-form representation of texts.

1.2 Distributed Representation of Words

Distributed representation [68] denotes linguistic terms (e.g., words, phrases, sentences) in dense numeric vectors. Figure 4 illustrates the distributed representation of five words in an example space, where every word is represented by a two-dimensional numeric vector. *Distributed* representation here means (1) every word is represented with multiple dimensions, and (2) every dimension is used to represent multiple words. The flexibility of distributed representation provides the possibility of represent similar words with similar vectors. As shown in Figure 4, if we still use the distance between vectors as the metric of dissimilarity between words, it is intuitive that words "*burger*" and "*donut*" are similar to each other comparing to "*gave*".

Researchers in NLP propose a number of methods to construct the distributed representation of words. A wide range of representation methods fall into a category

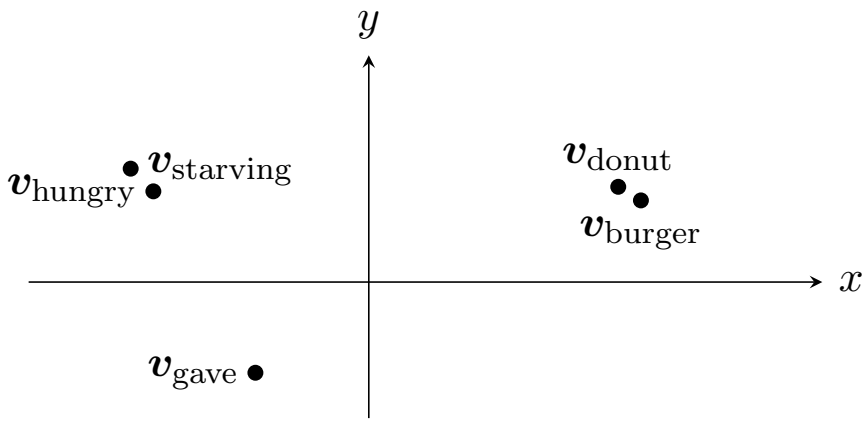


Figure 4: An illustration of the distributed representation of words in a 2-D space.

called *distributional representation*, which is in general a special case of distributed representation. It refers to any representation constructed from a method based on the *distributional hypothesis* [64]. The hypothesis says that words that occur in the same contexts tend to have similar meanings. In distributional representation, words that have the similar contexts will be represented by the similar numeric vectors.

1.3 Distributed Representation Functions for Texts

Example 2.

Consider the representation of the following two sentences with a linear representation function

(1) Bob gave Tina the burger.

(2) Tina gave Bob the burger.

For discourse processing, having a representation of words is not the end of the story, since the smallest unit in discourse processing is a piece of text including at least several words. The first question of applying distributed representation for discourse processing is *how to construct the distributed representation of texts, given the representation of words?* Consider the example sentence "*She was hungry*", the

problem is how to construct the vector $\boldsymbol{v}_{\{\text{She was hungry}\}}$ given $\boldsymbol{v}_{\text{She}}, \boldsymbol{v}_{\text{was}}, \boldsymbol{v}_{\text{hungry}}$,

$$\boldsymbol{v}_{\{\text{She was hungry}\}} = \boldsymbol{f}(\boldsymbol{v}_{\text{She}}, \boldsymbol{v}_{\text{was}}, \boldsymbol{v}_{\text{hungry}}) \tag{2}$$

where $\boldsymbol{f}(\cdot)$ is a mathematical function mapping the word representations to the sentence representation.

A special case is a linear representation function with addition as the composition operation. For example, the representation of the sentence "*She was hungry*" is

$$\boldsymbol{v}_{\{\text{She was hungry}\}} = \boldsymbol{f}(\boldsymbol{v}_{\text{She}}, \boldsymbol{v}_{\text{was}}, \boldsymbol{v}_{\text{hungry}}) = \boldsymbol{v}_{\text{She}} + \boldsymbol{v}_{\text{was}} + \boldsymbol{v}_{\text{hungry}} \tag{3}$$

This simple representation function $\boldsymbol{f}(\cdot)$ is useful in modeling phrases and short sentences in prior work [144, 16]. Prior work also demonstrates that word representation constructed with the distributional hypothesis [64] could encode semantic information [139] (Figure 4). In this thesis, I propose to learn word representation in the additional form, driven by the supervision information from discourse annotation. The simplicity of this representation function is crucial for a successful modeling when the annotated data is scarce.

However, the simplicity sacrifices the word order information in constructing sentence representation. Consider the two sentences in Example 2. The numeric representations with $\boldsymbol{f}(\cdot)$ are same, even though the meanings of them are totally different. The situation could be even worse for long sentences, because different composition orders of a long sentence may also end up with different meanings. To solve this problem, I propose to use artificial neural networks (ANNs) [14] to build the representation function. There are two advantages of using ANNs: (i) the nonlinearity of ANNs allows that a neural network based representation function is sensitive to word orders, therefore different sentences consisting of the same words (like the sentence (1) and (2) in Example 2) will eventually have different representations; (ii) with a neural network based representation function, we are able to develop a unified learning framework for both discourse processing and text representation.

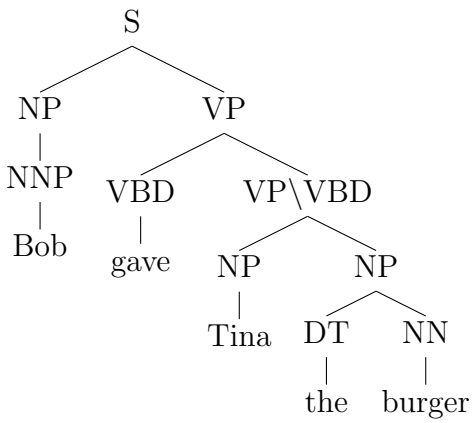


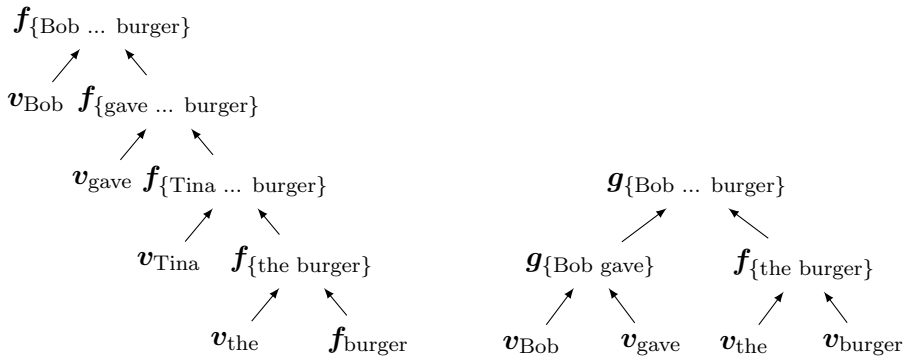
Figure 5: The syntactic structure (constituent parse) of the example sentence "*Bob gave Tina the burger*".

Figure 2(c) illustrates this new framework. Compare to Figure 2(b), I replace the feature extractor with a distributed representation function. Correspondingly, the outputs of this component will be the distributed representations of texts. More important, during learning, the supervision information from discourse annotation can backpropagate to the representation function, which allows the learning procedure to tune the the representation function together with discourse models.

In this thesis, I present two categories of the neural network based representation functions: the first category provides the possibility of incorporating rich linguistic features into a representation function; and the second category focuses more on modeling discourse information with a procedure of word generation. I will explain the linguistic intuitions behind these two model categories. To simplify the description, I assume each representation function is built on the sentence level. Extending these functions to a text including multiple sentences will be discussed in other chapters of this thesis.

1.3.1 Representation functions with rich linguistic features

The first idea of constructing a representation function is inspired by the principle of compositionality from formal semantics [157]: *the meaning of a sentence is a function*



(a) The representation function based the syntactic structure in Figure 5.

(b) The representation function of the same sentence but from Tina's perspective.

Figure 6: The representation functions with rich linguistic features.

of the meanings of the words within this sentence and of the way they are syntactically combined. Figure 5 shows the syntactic structure of combining words to form phrases and the entire example sentence. Starting from the bottom of this structure, it shows that combining the word "*the*" and "*burger*" to form the noun phrase "*the burger*". To build a representation function, the similar idea is employed to compose the distributed representations of word $\boldsymbol{v}_{\text{the}}$ and $\boldsymbol{v}_{\text{burger}}$ to obtain the distributed representation of phrase $\boldsymbol{f}_{\{\text{the burger}\}}$. The mathematical function $\boldsymbol{f}$ simulates a simplified semantic composition in formal semantics. As shown in Figure 6(a), the same function is used recursively to get the distributed representation of the example sentence. The neural network model in Figure 6(a) is called *recursive neural network*. Recently, it has been used for some NLP tasks relevant to either single sentence [175] or entire document [115]. In my work, I use it as a base model to represent texts with more linguistic features.

Example 3. *Consider the discourse relation between these two sentences*

(1) Bob gave Tina the burger.

(2) He *was hungry.*

To motivate which linguistic feature is also important for discourse processing, I make a little modification on the second sentence of Example 1. The new sentence is "*he was hungry*" (as in example 3). The difference is that the subject of the second sentence now refers to "*Bob*" not "*Tina*". In other words, the entity shared between two sentences is changed from *Tine* to *Bob*. The change on the entity reference also influences the discourse relation between two sentences. Instead of serving as a reason as in example 1, the modified second sentence provides a contrastive situation. This example illustrates that the discourse relation is not only about the semantic understanding of each sentence, but also about the entity coreference relation between sentences. To incorporate this linguistic phenomenon into distributed representation of sentences, I propose an additional recursive neural network structure to reflect the coreference relation between sentences. This additional structure (shown in Figure 6(b)) is an entity-oriented representation based on the syntactic structure in Figure 5. In this figure, $\boldsymbol{g}$ is another function to compose a node's parent node and its sibling node. Intuitively, it can be viewed as a representation of the sentence from an entity's perspective. The detail will be discussed in chapter 3.

1.3.2 Representation functions with rich generation power

The representation function category discussed in previous section directly targets on discourse processing. In this section, I would like to propose another category of representation functions from a different perspective: *how the discourse information could effect the word choice in language generation?*

Consider the example 4, the problem is to choose a word to complete the second sentence. The first sentence in this example provides the necessary contextual information for the second. Specifically, we know from the first sentence that "*the machine*" refers to "*MacBook Pro*". The first sentence also indicates the writer's negative opinion about MacBook Pro. However, the word "*But*" suggests the second

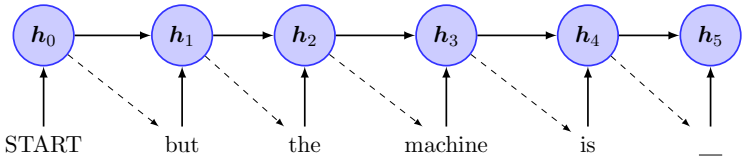


Figure 7: A RNNLM on a sentence from example 4.

sentence should describe something positive about this machine, in order to maintain the constrastive situation indicated by the word "*But*". This example shows the intuition on how discourse information could constrain the word choice to guarantee the coherence. In this section, I will briefly discuss how to formulate this intuition into a neural network model, and eventually get a representation function for discourse modeling.

Example 4. Choose a word to complete the following text:

Macbook Pro seems to be incompatible with everything else. But the machine is __.

The starting point of the mathematical formulation is a sentence-level language model called recurrent neural network language model (RNNLM) [137]. RNNLM is a probability distribution of words given the contextual information within the same sentence. Based on the probability distribution, RNNLM is able to predict the next most possible word. Figure 7 demonstrates the RNNLM on the second sentence of Example 4, where the solid arrows indicate the information flow and the dashed arrows indicate the word prediction. In each step, RNNLM makes a prediction of the next word based on the current state $\boldsymbol{h}$..

To introduce the discourse information into language modeling, we first need a context vector to summarize the contextual information from the preceding text. To simplify the model formulation, we only consider the contextual information from the previous sentence, and take last hidden states from the RNNLM of the previous

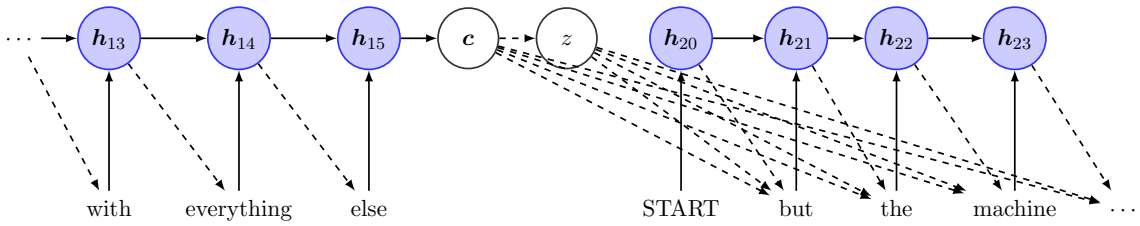


Figure 8: A language model incorporating with discourse relation and contextual information from previous sentence.

sentence as the context vector. For example, in Figure 8, we take $\boldsymbol{h}_{15}$ as the summarization of the contextual information for the first sentence in example 4, and rename it as $\boldsymbol{c}$. In the next step, context vector $\boldsymbol{c}$ is used to guess the possible discourse relation z with the following sentence. Then, both z and $\boldsymbol{c}$ are used to constrain the word choice, as shown by the dashed arrows in Figure 8.

The whole model is formulated in a generative procedure, where we start from the previous sentence to predict the discourse relation, then use the predicted discourse relation to help generate the words in the next sentence. For discourse modeling, the generative procedure can be reversed using Bayes' rule [15] to predict discourse relation z with the information from both sentences. In addition, the generative procedure can be directly applied to language modeling with discourse information. More detail about this model will be discussed in chapter 4.

1.4 Representation Learning with Distant Supervision

The proposed models in section 1.2 all assume that supervision information is available for discourse processing. However, the assumption is not always valid for some tasks. There are only a few corpora with human annotated discourse information. Most of them are news articles from the Wall Street Journal in the 1980s. For some discourse-relevant tasks, like discourse processing on academic writing, we may not have any annotated data. Therefore, it is imperative to explore the possibility of discourse processing with distant supervision.

Fortunately, discourse processing in some cases is much easier than in others. Consider the example 5, the discourse relation in both sentence (a) and (b) is CONTRAST. Because of the word "*but*" in example (a), it is easy to recognize the relation. In example (2), though, more semantic understanding on the sentence is required to identify the relation. The words like "*but*", which signal discourse relations explicitly, are called *discourse connectives*. Discourse connectives facilitate the relation identification task for the cases including them (called *explicit cases*). Is it possible to use supervision information from explicit cases to help predict the relations in the cases without any discourse connective (called *implicit cases*)? The idea of using explicit cases to help predict implicit cases is called *learning with distant supervision*.

Example 5.

Consider the discourse relations in the following two examples:

(1) Macbook Pro seems to be incompatible with everything else. But the machine is awesome.

(2) The key bindings of Macbook may take you a long time to get used to. Other parts of it are easy to use.

Prior attempts on discourse processing with distant supervision constantly present some negative results. For example, Sporleder and Lascarides (2008) [183] show that the discourse model learned from explicitly cases works poorly on the implicit cases. We argue that the major problem is representation method used in this work. Just like prior work on discourse processing, it used a surface-form representation with some manually-designed features. As discussed in section 1.1, it is not surprise about the inferior performance presented in [183].

To deal with this issue, I utilize a representation learning method to map the

surface-form representations of both explicit and implicit cases into a common low-dimensional space. This is just another way to construct the distributed representation. It share the same merit of other distributed representation forms — information learned from some cases can be easily transferred into other cases. More detail of this work will be discussed in chapter 5.

1.5 Applications of Discourse Processing

In this section, I will introduce how discourse information can be used to improve the performance of some NLP tasks, including document-level sentiment analysis and document-level machine translation. More detail is included in chapter 6.

1.5.1 Sentiment Analysis with Discourse Information

Sentiment analysis [155, 194] is a research area which analyzes the attitude or subjectivity of a text. The output of sentiment analysis is usually positive, negative or neutral. Most of existing work on sentiment analysis mainly focuses on the sentence level. A simple method of sentence-level sentiment analysis is counting the number of positive and negative words. If, for example, the positive words are more numerous than the negative words, the text has positive polarity. However, it will be problematic to directly apply a sentence-level sentiment model to long texts. As shown in Figure 1(a), the simple method considers the user review a neutral text.

Discourse structures of this review can help this simple method to give a more accurate result. Figure 1(a) illustrates how discourse structure can help to predict sentiment polarity. First of all, a simple sentence-level sentiment classifier is employed to justify the sentiment polarity of each sentence. Instead of treating every sentence as equally important, the discourse information identifies which sentences are more important with respect to the writer's intention. Then, the important sentences are assigned with a higher weight in computing the overall sentiment score of this review. In addition, the weight of each sentence can also be learned given the overall sentiment

score of a review and its discourse structure. Both ideas will be discussed in more detail in section 6.1.

1.5.2 Discourse-aware Machine Translation

Similar to sentiment analysis, state-of-the-art machine translation (MT) systems [99] also works on the sentence level only. Directly translating a text with multiple sentences is limited by its feasibility. For a given text in the source language (for example, Chinese), the set of translation candidates in the target language (for example, English) can be exponentially large with the number of tokens. Therefore searching for the optimal translation candidate (called *decoding* in MT) can be difficult or even impossible in a long text. However, translating a text sentence-by-sentence will definitely lose context information, which can be useful for decoding words in the current sentence. As shown in Figure 1(b), without context information, the translation for the second sentence looks inappropriate.

In this thesis, I present preliminary work on discourse for machine translation based on the joint models on discourse and word generation. Instead of running into the difficulty of decoding words from a long text directly, I assume that each sentence in the source language has been translated into the target language with up to n candidates. (In MT, the list of n candidates is often called a n-best list.) The task of document-level machine translation is to perform an additional decoding procedure on the n-best list given contextual information as additional features for decoding.

1.6 Contributions

As a summary of contributions, I explore the utility of representation learning for automatic discourse processing. The goal is to show the value of learning distributed representation in discourse modeling with supervision and distant supervision information. More specific, the benefit is demonstrated from three aspects: (1) supervised representation learning for discourse parsing; (2) context representation learning for

joint discourse and language modeling; (3) unsupervised feature representation learning for discourse processing with distant supervision. Each aspect represents a dimension of research on how representation learning can be used to help analyzing discourse automatically. The contributions of this thesis also include two application examples on how discourse information can be used to improve the performance on document-level NLP tasks. Detailed contributions consist of the following:

- with task-specific supervision, learning distributed representation is valuable for discourse processing (chapter 3). The work presents two possible ways to learning distributed representation for discourse units: (1) learning word embedding with simple composition operation, and (2) learning semantic composition with fixed word embedding. Particularly, the work on learning word embedding for RST parsing [81] is the first work introducing representation learning to discourse processing.

- show the way of incorporating discourse information into language modeling (chapter 4). The idea is based on the linguistic intuition that the context information from preceding text constrains the word choices in the current sentence. In this work, a recurrent neural network (RNN) language models concatenated with discourse relations is proposed to incorporate discourse information. This is the first work on combining discourse information for language modeling.

- demonstrate the possibility of discourse processing with distant supervision (chapter 5). The aim of using distant supervision in discourse processing is to utilize discourse relation pairs with discourse connectives (explicit cases) to help predicting those without discourse connectives (implicit cases). The proposed idea is using unsupervised feature representation learning to map all cases into a common latent space. Then, the discourse model learned in the explicit cases can be transferred into the implicit cases. The hope is to reduce the dependency

of discourse processing on annotated corpora. This work demonstrates that the idea of unsupervised representation learning makes the performance of distant supervision close to its fully-supervised counterpart. This was claimed to be impossible in prior work without representation learning [183].

- contribute to other relevant NLP tasks by utilizing discourse information (chapter 6). In this work, I present two methods of using discourse structures to help document-level sentiment analysis. Empirical evaluation shows that discourse information is critical for the document level, if we are only able to have a simple sentence-level sentiment classifier. I will also discuss the possibility to make use of context language models for document-level machine translation.

CHAPTER II

BACKGROUND

This chapter summarizes some relevant topics serving as the background information for the rest of the thesis. Starting from the discussion on discourse structure, it continues to include the basic information about discourse processing and the representation issue in discourse processing.

2.1 Discourse Structure

Researchers in computational linguistics share the opinion that *discourse has structure* [71]. Recognizing discourse structure is crucial for comprehending the discourse [87], interpreting coreferent mentions (e.g., pronouns) [70], identifying the temporal order of events [107], and understanding the main opinion from an argument. In addition, research on text generation also shows that organizing sentences with certain discourse structure guarantees the coherence of a generated text [209].

There has been a long-time debate about what the discourse structures should look like and how to characterize them. Each theory proposes to explain the coherence of discourse from a specific perspective. In general, there are five ways to characterize discourse structure in computational linguistics [147]. In this section, we would like to shed light on three (Intentional structure, Attentional structure and Rhetorical structure) that are closely relevant to the recent progress of discourse processing, and leave the Information structure and Informational structure for a further reading [147].

Intentional structure focuses on describing a writer communicative plan with a coherent text. Intentions encode what the writer was trying to accomplish with discourse. The intuitive explanation of the intentional structure is that every discourse

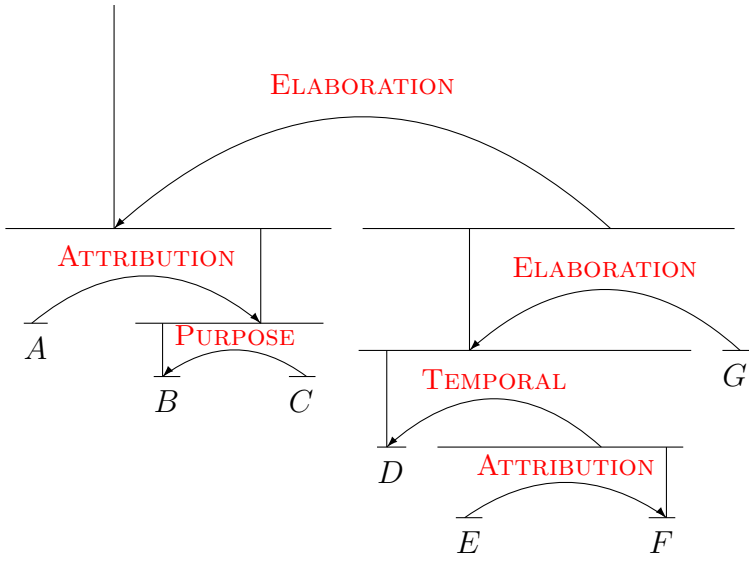
has its purpose. This purpose plays a fundamental role in characterizing the discourse structure [59]. On the other hand, attentional structure characterizes discourse structure from a reader's perspective. Instead of reflecting the writer's mental state, **attentional state** contains information about the objects, properties, relations, and discourse intentions that are most salient at any given point in discourse [59]. To understand a discourse, a reader usually focuses the attention on a small set of entities and shifts the attention to new entities along with the reading or conversation.

Rhetorical structure is used by many researchers in computational linguistics to explain a wide range of discourse phenomena. The basic idea of a rhetorical structure is that discourse structure is characterized by a set of rhetorical relations (*or* discourse relations, or coherent relations). Researchers in interpretation have argued that recognizing these relationships is crucial for explaining discourse coherence, resolving anaphora, and computing conversational implicature [72].

Many theories on rhetorical structure have been proposed in the past thirty years on discourse research. Each is accompanied by a set of pre-defined rhetorical relations associated with the theory. In the rest of this section, I briefly introduce two theories of rhetorical structures, which have dominated computational discourse processing for more than a decade. In addition, for each theory, I will discuss the benchmark datasets for computational study — the availability of these datasets helps to popularize the corresponding theory in computational linguistics.

2.1.1 Rhetorical Structure Theory

Rhetorical Structure Theory (RST) is a descriptive linguistic approach to a range of phenomena in the organization of discourse. The theory is based on a few assumptions about how written text functions, and how it involves words, phrases, grammatical structure, or other linguistic entities [125]. RST addresses text organization by means of *relations* that hold between parts of a text. It explain coherence by postulating a



[Shearson Lehman Hutton Inc. said]^{A} [it applied to Taiwanese securities officials for permission]^{B} [to open brokerage offices in Taipei.]^{C} [Shearson's application is the first]^{D} [since the Taiwan Securities and Exchange Commission announced June 21]^{E} [that it would allow foreign brokerage firms to do business in that country.]^{F} [Taiwan officials are expected to review the Shearson application later this year.]^{G}

Figure 9: An example of RST from the RST-DT (adapted from the RST-DT `wsj_0639).`

hierarchical, connected structure of texts, in which each part of a text has a role, a function to play, with respect to other parts of the text. The notion of text coherence through text relations is widely accepted, and these relations have also been called *coherence relations* [70], *discourse relations* [98] or *rhetorical relations* [1].

An example RST tree is illustrated in Figure 9. The whole text in this tree structure is divided into seven text units, $\{A, B, \ldots, F, G\}$. In RST, they are called elementary discourse units (or EDUs). The tree is built by connecting two adjacent text units using discourse relations in a bottom-up fashion. Consider the relation between discourse unit B and C. The arrow from unit C to B indicates that B is the **nucleus** and C the **satellite** within this discourse connection. In RST, the nucleus is considered to be more important than the satellite. In other words, B is more essential to the writer's intention, and C is to supportive information that is

often incomprehensible without its nucleus. The discourse relation between B and C is PURPOSE, which is a relation type defined in the classical RST paper by Mann and Thompson [124]. According to Mann and Thompson [124], the satellite in the PURPOSE relation (C in this case) presents a situation to be realized through the activity in the nucleus (B in this case).

RST Discourse Treebank (RST-DT) [23] is a corpus annotated with the RST framework. It includes 385 documents from the Wall Street Journal (WSJ) corpus annotated with RST structures similar to Figure 9. The discourse relations in the RST-DT are extended to 78 fine-grained discourse relations within 18 coarse-grained relation classes. Annotating the RST-DT was not an easy task. Even though every human annotator is a well-trained linguistic expert on RST, the inter-annotator agreement on the RST-DT is still very low. For example, the agreement on the 18 coarse-grained relation classes is less than 66%.

2.1.2 Lexicalized Tree Adjoining Grammar for Discourse

Unlike RST, which builds a high-level discourse structure on a text, Lexicalized Tree Adjoining Grammar for Discourse (D-LTAG) [204, 52] provides an intermediate level of discourse structure on the clauses within a text. It is motivated by the observation that discourse connectives function syntactically and semantically in the discourse akin to the way verbs function in the clause. For example, in LTAG, verbs are viewed as *predicates*, which supply relations between entity interpretations of noun phrases or clauses. D-LTAG borrows the similar structure from LTAG and applies it to the discourse level. In D-LTAG, a predicate is a discourse connective. It can be a subordinating conjunction (e.g.; "*because*", "*when*", etc.), coordinating conjunction (e.g.; "*and*", "*or*", etc.), and discourse adverbial (e.g.; "*for example*", "*instead*", etc.). It also includes modified and conjoined forms of discourse connectives; such as *only because*, *if and then*. An argument of the predicate can be individual clauses or

`Explicit Relation` : **Shearson's application is the first** since *the Taiwan Securities and Exchange Commission announced June 21 that it would allow foreign brokerage firms to do business in that country*

`Implicit Relation` : **the brokerage service will be directed at individual investors who want to buy foreign and domestic stocks** [also] *It's an attractive market with good growth opportunities*

Figure 10: Examples from the article `wsj_0639 in the PDTB.`

complex units composed of clauses and sentences.

The Penn Discourse Treebank (PDTB) [165] is the corpus annotated with the D-LTAG. In the PDTB, several categories of discourse relations are distinguished by how they are realized in text. In Example 10, two major categories — **explicit** and **implicit** categories — are demonstrated. The *explicit* category refers to the cases where relations of argument pairs are signaled by discourse connectives explicitly. For instance, the relation between the two clauses in the first example in Figure 10 is signaled by the discourse connective "*since*". On the contrary, the *implicit* category refers to cases without any discourse connective. For implicit cases, discourse connectives are inferred from the text, as shown in the second example in Figure 10. Some other categories, including *AltLex*, *EntRel* and *NoRel*, are used for cases where neither explicit nor implicit categories are applicable.

In the PDTB, discourse relations are assumed to hold between two and only two arguments. The two arguments of a connective are simply labeled as `Arg1` and `Arg2`, because there are no generally accepted abstract semantic categories for classifying the arguments (like; *agent* or *patient* suggested by verbs in semantic role labeling), For explicit type, `Arg2` is the argument syntactically bounded with the connective, while `Arg1` is the other argument. In addition, arguments of an explicit relation are not constrained to be single clauses or sentences. As shown in the PDTB, an argument could range from a clause to a sentence, or even multiple sentences.

For both explicit and implicit categories of relations, the PDTB provides a three-level hierarchy of discourse relations. The first level consists of four major relation **classes**: TEMPORAL, CONTINGENCY, COMPARISON and EXPANSION. For each class, a second level of **types** is defined to provide finer semantic or pragmatic distinctions; there are sixteen such relation types. A third level of **subtypes** is defined for only some types, specifying the semantic contribution of each argument. More information about the PDTB is provided by Prasad *et al.* [165].

2.2 Computational Models for Discourse Processing

This section introduces basic information of computational models for discourse processing. Those models are either machine learning models or use machine learning models as essential components. In this section, I informally treat a machine learning model as a black-box system which makes predictions based on its inputs. More detail about learning will be provided in section 2.3.

2.2.1 RST-style discourse processing

The pipeline of RST-style discourse processing is illustrated in Figure 11 Automatic processing on the RST-DT involves two steps: (i) **discourse segmentation** and (ii) **discourse parsing**. Given a document, discourse segmentation is to divide the document into a set of EDUs. The basic formulation of discourse segmentation is, given a token, to determine whether it is an EDU boundary. This decision problem can be solved with a binary classifier by taking this token and its surrounding as input [171] (more detail about classification models will be discussed in section 2.3). Hernault *et al.*obtained even better results with a sequential labeling model [66]. Based on sequential modeling, Bach *et al.*[2] were able to get a further improvement up to 95% F_1 measure with a re-ranking trick. (Compare to the 98% F_1 measure on human annotation agreement of discourse segmentation.) I would like to focus on the most challenging part of RST-style discourse processing — discourse parsing.

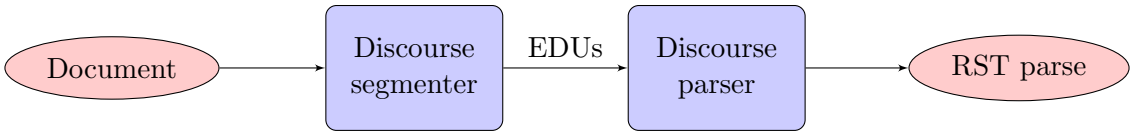


Figure 11: The system pipeline of RST-style discourse processing.

The basic parsing procedure includes two tasks: given adjacent text units, a parser needs to (1) identify whether these two units have a discourse relation; (2) if the answer is positive, then identify the discourse relation and label each unit as either nucleus or satellite[1]. Prior work on discourse parsing shows three ways to build a RST tree:

Two-step classification is proposed by Hernault*et al.* [66]. The first step is to scan the whole set of discourse units sequentially, and find two most relevant adjacent units; the second step is to determine the discourse relation between these two units and connect them with the predicted relation to form a larger discourse unit. Then, continue these two steps until the whole set of EDUs are all connected as a single discourse span. The complexity of this algorithm is $O(N^2)$, where N is the number of EDUs.

Shift-reduce parsing is linear parsing algorithm with respect to N. The parsing procedure consists of a sequence of either *shift* or *reduce* parsing actions. A shift-reduce parser maintains a stack and a queue to keep record of parsing status. Initially, all discourse units (in this case, EDUs) are in the queue. A shift action fetches a unit from the head of the queue and pushes it into the stack. A reduce action pops two discourse units from the stack, combines them together with a discourse relation, and then pushes the larger unit back to the stack. Given certain parsing status, the next parsing action usually is determined by a multiclass classifier [129, 171]. Each parsing action is the optimal decision

[1]In RST-DT, all text units can be nuclei.

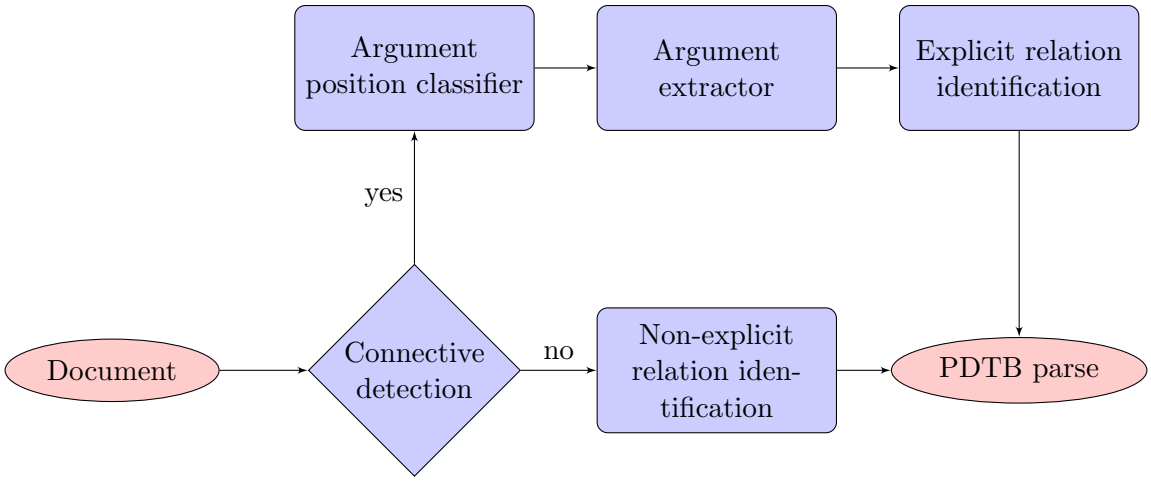


Figure 12: The system pipeline of automatic RST-style discourse processing.

based on the current status, so it is a local optimal algorithm. The algorithm will end with only a discourse unit in the stack. In total, there are $2N - 1$ parsing actions.

Cocke-Younger-Kasami parsing is first proposed by Joty *et al.*[89, 90] for sentence-level and document-level discourse parsing. The motivation of using CKY parsing algorithm is to obtain a global optimal parse. For each possible RST tree constituent, a Conditional Random Field (CRF) model [104] is used to compute the probability. Given the probability of all possible constituents, the CKY parsing algorithm finds the most possible candidate using a bottom-up fashion [91]. The complexity of this algorithm is $O(N^3)$.

2.2.2 PDTB-style Parsing

The system pipeline of an end-to-end PDTB parsing is shown in Figure 12. This pipeline is adopted from [119] with a little modification for illustration purpose. In practice, a PDTB parsing system usually employs two-pass processing for parsing: for a given document, the first pass is to detect all existing discourse connectives for explicit cases; then the second pass processes the rest of the document for non-explicit cases.

If a discourse connector is detected, the system will continue to identify the positions of two arguments and then extract both arguments. As discussed in the previous section, the argument (especially `Arg1`) position of an explicit relation can be arbitrary, so it is necessary to identify the position before argument extraction. Extraction of `Arg1` for a discourse connective is also not trivial, as it can be either a clause or multiple clauses. Actually, argument labeler (including position identification and argument extraction) turns out to be the difficult part of PDTB-style parsing in explicit relation cases [119]. The last step of explicit relation identification is relatively easy [163]. As reported by Pitler [163], the classification accuracy on the first-level PDTB relations is 93.09%.

The situation is totally different for non-explicit cases, in which no discourse connective is detected between two adjacent sentences. Non-explicit types include EntRel, NoRel, AltLex and Implicit. Of interest is the implicit type, where two adjacent sentences have a semantic/pragmatic connection, but no discourse connective is used explicitly. As in implicit cases, `Arg2` always follows `Arg1`, which makes argument labeling and extraction a straightforward task. The major difficulty is implicit discourse relation identification, due to the missing discourse connective between arguments [118, 162]. Without any discourse connective, identifying a discourse relation requires more semantic understanding of texts.

Although inferring discourse connectives first may help relation prediction as a final task [213], most of the prior work on implicit relation identification tries to predict discourse relations directly without the inference of discourse connectives [118, 156, 105, 13]. Discourse relation prediction is usually a straightforward classification problem (section 2.3). The basic classification setup is, given an argument pair, a classifier is employed to predict the discourse relation holding between them. A critical issue in relation prediction is to represent arguments in an appropriate way, so the representation could encode some semantic information for classification. We

will discuss the representation issue and some relevant work in section 2.4.

There are two approaches to perform a prediction task on implicit discourse relation identification. The first approach is multiclass classification [118], which requires identifying the discourse relation from all possible choices. More recent work has emphasized binary classification, where the goal is to build and evaluate separate "one-versus-all" classifiers for each discourse relation [162, 156, 13]. Multiclass classification is more relevant to the practical purpose, for example, to build a discourse processing system. In this thesis, I consider both of them as evaluation for my discourse processing systems.

2.3 Machine Learning Models of Discourse Processing

To make the thesis self-contained, this section will introduce some basic information about machine learning particularly relevant to discourse processing. Some textbooks [15, 148] and notes [46] are good resources for further reading.

2.3.1 Discourse processing as classification

As discussed in section 2.2, most of the discourse processing tasks can be solved directly as a classification problem, or else use classification models as an essential component. Here are some typical examples:

- Discourse segmentation in RST-style processing: A binary classifier [171] is employed to determine whether a token is on an EDU boundary.
- RST-style parsing with shift-reduce algorithm: A multiclass classifier [81] is used to select the optimal parsing action (e.g., *shift*, *reduce* with a certain nucleus-satellite structure and relation type), based on the status of parsing queue and stack.
- PDTB-style parsing [119]: A multiclass classifier is used to identify discourse relations holding in argument pairs.

Once a task is expressed as a classification problem, the next step is building a classification model to solve this problem.

A classification model can be formulated with the following form either *explicitly* or *implicitly*[2],

$$y_c = \boldsymbol{w}_c^\top \boldsymbol{f}(\boldsymbol{x}), \tag{4}$$

where $\boldsymbol{x}$ is the input that depends task we would like to solve, $\boldsymbol{f}(\boldsymbol{x})$ is the feature representation function based on input[3], $\boldsymbol{w}_c$ is the classification weight associated with the class label c, and y_c as the dot product of $\boldsymbol{w}$ and $\boldsymbol{f}(\boldsymbol{x})$ is called decision value. . Consider the relation identification task in the PDTB, where $\boldsymbol{x}$ will be an argument pair and c is a specific discourse relation defined in the PDTB. To make a decision based the classification model, we pick the label c which has the maximal decision value, $\hat{c} = \arg\max_c y_c$.

Feature representation function $\boldsymbol{f}(\boldsymbol{x})$ maps the input $\boldsymbol{x}$ into a numeric vector. For the relation identification task, a simple $\boldsymbol{f}(\boldsymbol{x})$ could just be the bag-of-words representation of $\boldsymbol{x}$, and a complicated $\boldsymbol{f}(\boldsymbol{x})$ could be a feedforward neural network with multiple layers. How to represent texts is an important issue of applying machine learning for discourse processing [8], since we always need to reduce the gap between what we have (plain texts) and what we want (discourse relations in texts). Section 2.4 of this chapter discusses the representation issue for discourse processing.

Training a classification model involves learning the weight vector $\boldsymbol{w}$ defined in Equation 4. Given a training example $(\boldsymbol{x}, c)$, training a model is equivalent to optimizing a pre-selected objective function (e.g., cross entropy [148] or max-margin loss [168]) defined on the decision function. There are many ways to optimize the objective functions. A simple but effective method is stochastic gradient decent (SGD), which iteratively computes the gradient of the loss function ℓ as $\frac{\partial \ell}{\partial \boldsymbol{w}}$, then update $\boldsymbol{w}$

[2]For example, a nonlinear SVM [15] is formulated in this way implicitly with a kernel function.

[3]Another way to formulate the decision function in NLP is $y_c = \boldsymbol{w}^\top \boldsymbol{f}(\boldsymbol{x}, c)$ [46], while in this way $\boldsymbol{f}$ should not be called as a representation function.

with the gradient as

$$\boldsymbol{w} \leftarrow \boldsymbol{w} - \eta \frac{\partial \ell}{\partial \boldsymbol{w}} \tag{5}$$

where η is the learning rate. More discussion about training with SGD can be found in [20].

Besides learning the weight $\boldsymbol{w}$, we can also learn the representation function $\boldsymbol{f}(\boldsymbol{x})$ by formulating it as a mathematical function with parameters, for example, a neural network model. The idea of learning feature representation function is called **representation learning** [8] in the machine learning community. The motivation of using representation learning is to reduce the dependence of a machine learning model on feature engineering, and to extract the task-relevant information from data effectively. In section 2.4, I will discuss the importance of representation learning from linguistic perspective.

2.4 Representation for Discourse Processing

All representations for discourse processing can be classified into three categories: logical representations, surface-feature representations, and distributed representations. A classical work on the logical representation for discourse processing is from Hobbs [70]. In his work, each sentence is represented as a set of logical propositions, and inference on discourse relations relies on a huge knowledge base [152]. In the rest of this section, I will summarize two other representation methods on discourse processing: surface-form representation and distributed representation.

2.4.1 Surface-form Representations

Due to the difficulty of using logical representations, most recent work on representation for discourse processing falls into this section. As shown in Figure 2(b), the unified framework is a learning based discourse model on top of a feature extractor. Different tasks may need different learning models. In this section, I summarize the important surface features in discourse processing into several categories. I use

sentences as the minimal processing units for the convenience of description in this section, which should be replaced by EDUs in the RST-DT, and by arguments in the PDTB.

Lexical features The category includes the features extracted from lexical level, for example, n-gram features [90], words in the beginning or end of sentences [171], or even cue phrases from a given list [129, 97]. Lin *et al.* [118] found that for implicit related identification on the PDTB, word tuples extracted from the sentence pairs (such as $\langle \text{burger}, \text{hungry} \rangle$) are also useful. In discourse processing, lexical features are usually used as baseline features. They can be useful — observing some lexical features (e.g., discourse connectives) could make relation prediction a relatively easy task.

Syntactic features This category contains the features extracted from the syntactic structures of sentences, such as production rules from constituent parses and dependency pairs from dependency parses [118]. In addition, dominance sets [182, 66] are also claimed to be effective features for intra-sentential parsing in the RST-DT.

Semantic features Surface features can also represent certain semantic information with help from external resources. WordNet is one of the most popular resource for measuring semantic similarity between words. It has been used for constructing features from the 1990s [129]. Levin verb class [113] is widely used in the implicit relation identification on the PDTB [162, 156, 13, 105]. The intuition of using Levin verb class is that the relation between arguments is more like to be EXPANSION if the verbs from two arguments are from the same verb class, e.g., the verb "*lend*" and "*loan*" belong to the `Give` verb class. Recently, Rutherford and Xue [170] show that using word Brown cluster pairs instead of word pairs directly could also improve the performance on predicting implicit relations on the PDTB. There are also some

other lexical semantic resources. For example, Multi-perspective Question Answering (MPQA) subjectivity lexicon [206] can help to capture the sentiment polarity of arguments [156, 162], which is useful for the discourse relations like CONTRAST. Even so, it is worth to understand the limitations of using surface-form semantic features. First, these features heavily depend on the existence of external resources. Second, like other surface features, feature representation cannot be refined with learning signals.

Contextual features These are used to indicate the interaction with local context. Lin *et al.* [118] believes that the overlap of arguments could benefit implicit relation identification. The same idea is shared by Feng and Hirst [50] in their work on RST-based discourse parsing. Furthermore, Ghosh *et al.* [53] extend a similar idea into a more broad scope, where each time five sentences are considered as context for feature extraction.

Structural features Most of structural features are used in RST parsing. The purpose is to indicate position of a text span with respect to the entire document. These features include the distance between an EDU and the beginning (*or* end) of the document [90], etc. The category also includes some features denoting structures indirectly. For example, given an EDU from a document, what are the last several parsing operations performed on the document [129].

Entity-based features For implicit relation identification on the PDTB, Louis *et al.* [122] attempt to predict the relations using some entity-based features, such as grammatical role, syntactic realization etc. Though the results are preliminary, they conjecture that some other aspects of entity based features may be useful. Recently, Rutherford and Xue [170] illustrate that, on the same problem, some simple coreferent pairs of entities can be used as additional features. Entity-based features are also

used for RST-based discourse parsing. An example is the entity-transition features constructed on the document level [50].

All the features explained in this section fall into the category of surface-form feature representation. The problems of using surface-form have been explained in section 1.1.

2.4.2 Distributed Representation

Distributed representation was first proposed by Hinton *et al.*[68] in the classical monograph of artificial neural networks (ANNs): *Parallel Distributed Processing*. In the language of ANNs, it refers that each entity is represented by a pattern of activity distributed over many computing elements, and each computing element is involved in representing many different entities. It was proposed to overcome the computational limitation of **local representation** [68], where each entity is represented by a specific computing element. The benefit of using distributed representation, compared to local representation, is that "*it leads to automatic generalization*" (Hinton, 1986) [67].

Conceptually, both local representation and distributed representation have their counterparts in NLP. In the scenario of representing words as numeric vectors, a popularly-used local representation in NLP is called the *one-hot* representation, where every word is represented by a long and sparse vector, with only a single non-zero element. An example is the representations of words in Figure 3. A special case of distributed representation in NLP is *distributional* representation. The term *distributional* emphasizes that the representation is constructed based on Distributional Hypothesis [64]. The realizations of distributional representation include Latent Semantic Analysis (LSA) [44]) and its extensions, such as Latent Dirichlet Allocation (LDA) [17].

Distributed representation learning is to customize distributed representation based on context [44, 138] or driven by a specific task [123, 94]. Our work (Ji and Eisenstein,

2014) [81] is the first published work on representation learning for discourse parsing. Driven by the task of RST-style parsing, the distributed representation of words does encode certain amount of semantic information for relation identification. More technical detail on different representation functions will be explained in chapter 3.

2.4.3 Distributed Representation for Sentences

Although distributed representation has only been exploited in discourse processing very recently, the idea of distributed representation has been introduced into the NLP community for more than a decade. In 1991, Berg [11] put forward an idea of learning a representation through the linguistic structures. The intuition is based on the fact that the linguistic structure (e.g., syntactic structures) helps constrain the semantics or meaning of the sentence. Four years later, Goller and Kuchler [57] present a learning algorithm, called *backpropagation through structure* (BPTS). This algorithm makes it possible learning with structures. Starting from 2010, Socher and his colleagues introduce several variants of recursive neural networks. For applications, they apply these models to different sentence-level NLP tasks [178, 177, 175, 174, 175, 176]. All these works illustrate that composition through syntactic structures is a powerful method on modeling sentences. My work [82] extends this idea to the task on the cross-sentence level and also investigate the possibility of incorporating more linguistic features (section 3.2).

As discussed in chapter 1, I propose an additional composition pass to build the sentence representation from the entities' perspective (section 3.2). The idea of two-pass composition is also useful on some sentence-level tasks. Consider sentiment analysis on Example 6 [159]: the sentiment polarity on "*Android*" is definitely different with "*iOS*". Noticing this, Paulus *et al.* [159] introduce global belief recursive neural networks (GB-RNNs) to generate two distributed representations on every word. One representation is from the forward propagation step, the other is from

the backward propagation step. For every word, the forward representation is the distributed representation of word itself, and the backward representation includes the information propagated back from the sentence (root node of the parse tree). For sentiment analysis, both representations are used to predict the sentiment score.

Example 6. *Android beats iOS*

In addition, inspired by bi-directional recurrent neural networks, Irsoy and Cardie [76] present a bi-directional architecture on recursive neural networks. For their task of opinion expression extraction, two representations on the same node contain the information from the subtree with it as root and the information from the rest of the parse tree. Even the intuition is similar, Paulus *et al.* [159] have different methodologies on composing information.

The last relevant work is the inside-outside recursive neural networks from Le and Zuidema [109]. Their model shares the same composition procedure as my proposed representation functions. But the motivation and the applications are different. In their work, the goal is to support an infinite-order generative model of dependency parsing. In my case, the purpose is to induce the representations of the entities shared between two adjacent sentences.

CHAPTER III

DISCOURSE PARSING WITH SUPERVISED REPRESENTATION LEARNING

Work described in this chapter was undertaken in collaboration with Jacob Eisenstein, and published at ACL 2014 [81] and TACL 2015 [82]

In section 1.2, I explain the motivation of supervised representation learning for discourse processing. In this chapter, I will discuss the more detail three methods proposed for representation learning with supervision. The high-level idea is inspired by the success of my work on paraphrase detection (Ji and Eisenstein, 2013) [80]. In that work, a weighted matrix factorization [60] is employed on the bag-of-feature matrix to construct distributed representations of sentences. The feature weights are computed with a information-theoretical metric (KL-divergence) to measure whether the features are informative on detecting paraphrase. Therefore, learning weights requires the supervision information from an annotated corpus, for example the Microsoft Research Paraphrase Corpus (MSRPC) [42]. With the representation learned with supervision information, the model was able to obtain the state-of-the-art performance on the MSRPC. The success of this work points out a promising way of incorporating supervision information in learning distributed representation. The idea is further extended in this chapter for discourse processing. Instead of using the supervision information independently with learning distributed representation, a unified framework (as illustrated in 2(c)) is utilized to learn representation together with final tasks.

The rest of this chapter is outlined as follow: section 3.1 discusses the model of learning word embeddings for RST parsing. The semantic composition operation used

in this model is just simple addition. Then, section 3.2 moves to a deep discussion on distributed representation with semantic composition. In this section, recursive neural networks are considered for semantic composition, and also an additional composition operation for capturing the semantic connection across sentences. Section 3.3 presents a discussion based on the models in this chapter.

3.1 Distributed Representation Learning for RST Parsing

Discourse relation identification is a major task in discourse parsing. While recent work has introduced increasingly powerful features [49] and inference techniques [90], discourse relations remain hard to detect. Prior work on feature engineering with surface and syntactic features (as discussed in subsection 2.4.1) are not capable of capturing what are fundamentally semantic distinctions, particularly in the face of relatively small annotated training sets, like the RST Discourse Treebank [23].

In this section, I discuss the representation learning with linear functions for RST-style discourse parsing. The core idea of this 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. Several alternative representation function are considered for transforming the original features, corresponding to different ideas of the meaning and role of the latent representation. Alternatively, this approach can be seen as a non-linear learning algorithm for incremental structure prediction, which overcomes feature sparsity through effective parameter tying.

The parsing model is implemented with the shift-reduce parsing algorithm [129, 171]. This model is learned with a large-margin transition-based structure prediction [191]. At the same time, a linear representation function is learned jointly 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, visualization on 2-D space also shows that the distributed representation coheres well with the characterization of discourse connectives in the Penn Discourse Treebank [165].

3.1.1 Model

Recall that, the core idea is to project lexical features into a latent space that facilitates discourse parsing. In this way, the parsing model can capture the meaning of each discourse unit, without suffering from the very high dimensionality of a surface representation. While such representation learning approaches have been proven to increase robustness for parsing, POS tagging, and NER [142, 101, 193], they would seem to have an especially promising role for discourse, where training data is relatively sparse and ambiguity is considerable. Prasad *et al.*[166] 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, the 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. Specifically, a simple transformation function, linear projection, is considered in this work. Thus, the proposed approach is named DPLP: Discourse Parsing from Linear Projection.

Shift-reduce discourse parsing As mentioned before, the shift-reduce parsing algorithm, as first proposed by [129], is used for discourse parsing. At each point in the parsing process, the algorithm maintains 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. The parser can then choose either to *shift* the front of the queue onto

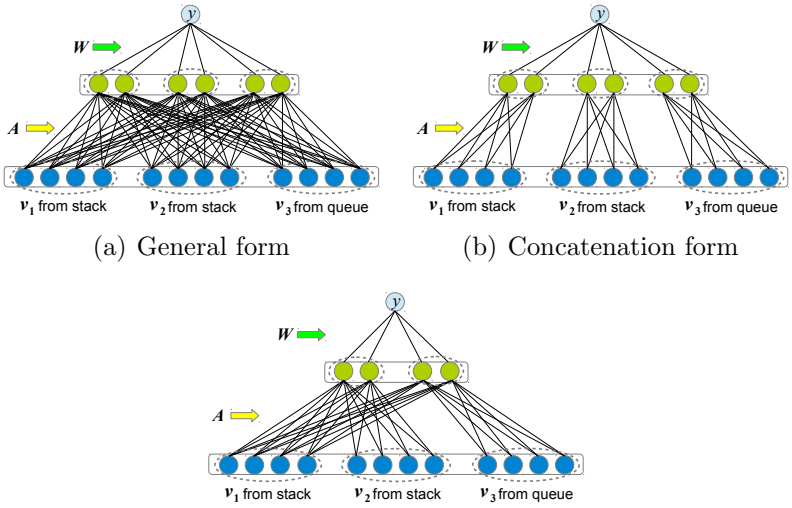
the top of the stack, or to *reduce* the top two elements on the stack in a discourse relation. The reduction operation must choose both the type of relation and which element will be the nucleus. Therefore, there are multiple reduce operations with specific relation types and nucleus positions. Shift-reduce 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 [129] and the averaged perceptron [33, 171] for this purpose. Instead, this work employs a large-margin classifier, because it allows to compute derivatives of the margin-based objective function with respect to both the classifier weights as well as the projection matrix.

Let $\mathcal{V}$ denotes the surface feature vocabulary, and $\mathbf{v} \in \mathbb{N}^V$ represents each EDU as the numeric vector, where $V = \#|\mathcal{V}|$ and the n-th element of $\mathbf{v}$ is the count of the n-th surface feature in this EDU. During shift-reduce parsing, we consider features of three EDUs: the top two elements on 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 $\mathbf{v} = [\mathbf{v}_1; \mathbf{v}_2; \mathbf{v}_3]$. The decision function based on the concatenated vector is formulated as

$$\hat{m} = \arg \max_{m \in \{1,\ldots,C\}} \mathbf{w}_m^\top \mathbf{f}(\mathbf{v}; \mathbf{A}) \tag{6}$$

where $\mathbf{w}_m$ is the weight for the m-th class and $\mathbf{f}(\mathbf{v}; \mathbf{A})$ is the *representation function* parameterized by $\mathbf{A}$. The score for class m (in this 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 19. Note that, after applying a reduce operation, the stack will include a span that contains multiple EDUs. This works follows the *strong compositionality criterion* of [128] and considers only the nuclear EDU of the span.

The representation function $\mathbf{f}(\mathbf{v}; \mathbf{A})$ can be defined in any form; for example, it could be a nonlinear function defined by a neural network model parameterized by



(a) General form (b) Concatenation form

(c) Difference form

Figure 13: Decision problem with different representation functions

A. Instead, this work focuses on the linear projection,

$$\mathbf{f}(\mathbf{v}; \mathbf{A}) = \mathbf{A}\mathbf{v}, \tag{7}$$

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 function can be rewritten as $\tilde{\mathbf{w}}_m^\top \boldsymbol{v}$, 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.

Three different constructions are considered for the projection matrix $\mathbf{A}$.

- *General form*: The general case places no special constraint on the form of $\mathbf{A}$.

$$\mathbf{f}(\mathbf{v}; \mathbf{A}) = \mathbf{A} \begin{bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \\ \mathbf{v}_3 \end{bmatrix} \tag{8}$$

  This form is shown in Figure 13(a).

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

$$\mathbf{f}(\mathbf{v}; \mathbf{A}) = \begin{bmatrix} \mathbf{B} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{B} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{B} \end{bmatrix} \begin{bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \\ \mathbf{v}_3 \end{bmatrix} \tag{9}$$

  This form transforms the representation of each EDU separately, and does 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.* The difference form explicitly represents the *differences* between adjacent EDUs, by constructing $\mathbf{A}$ as a block difference matrix,

$$\mathbf{f}(\mathbf{v}; \mathbf{A}) = \begin{bmatrix} \mathbf{C} & -\mathbf{C} & \mathbf{0} \\ \mathbf{C} & \mathbf{0} & -\mathbf{C} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \\ \mathbf{v}_3 \end{bmatrix}, \tag{10}$$

  The result of this projection is that the latent representation has the form $[\mathbf{C}(\boldsymbol{v}_1 - \boldsymbol{v}_2); \mathbf{C}(\boldsymbol{v}_1 - \boldsymbol{v}_3)]$, 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 $(V + 2C)\frac{K}{3}$.

3.1.2 Large-Margin Learning Framework

A large-margin prediction approach is applied to train the model. There are two parameters that need to be learned: the classification weights $\{\mathbf{w}_m\}$, and the projection matrix $\mathbf{A}$. 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, a learning procedure arrives at a saddle point of the objective function.

Mathematically, the learning objective can be formulated as the following constrained optimization problem,

$$\begin{aligned} &\min_{\{\mathbf{w}_{1:C},\xi_{1:l},\mathbf{A}\}} \frac{\lambda}{2}\sum_{m=1}^{C}\|\mathbf{w}_m\|_2^2+\sum_{i=1}^{l}\xi_i+\frac{\tau}{2}\|\mathbf{A}\|_F^2 \\ &\text{s.t. } (\mathbf{w}_{y_i}-\mathbf{w}_m)^\top \mathbf{f}(\mathbf{v}_i;\mathbf{A}) \geq 1-\delta_{y_i=m}-\xi_i, \\ &\qquad \forall\ i,m \end{aligned} \tag{11}$$

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 [35], the problem defined in Equation 11 can be solved via Lagrangian optimization:

$$\begin{aligned} &\mathcal{L}(\{\mathbf{w}_{1:C},\xi_{1:l},\mathbf{A},\eta_{1:l,1:C}\}) = \\ &\quad \frac{\lambda}{2}\sum_{m=1}^{C}\|\mathbf{w}_m\|_2^2+\sum_{i=1}^{l}\xi_i+\frac{\tau}{2}\|\mathbf{A}\|_F^2 \\ &\quad +\sum_{i,m}\eta_{i,m}\Big\{(\mathbf{w}_m^\top-\mathbf{w}_{y_i}^\top)\mathbf{f}(\mathbf{v}_i;\mathbf{A})+1-\delta_{y_i=m}-\xi_i\Big\} \\ &\quad \text{s.t. } \eta_{i,m}\geq 0\ \forall i,m \end{aligned} \tag{12}$$

Then, optimizing $\mathcal{L}$ 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 $\mathbf{f}(\mathbf{v}_i; \mathbf{A})$. In this work, the weights $\{\mathbf{w}_{1:C}\}$ and dual variables $\{\eta_{1:l,1:C}\}$ is learned from a standard dual-form SVM solver. Then, the learning procedure then updates $\mathbf{A}$, recomputes $\{\mathbf{w}_{1:C}, \eta_{1:l,1:C}\}$, and iterates until convergence. This iterative procedure is similar to the latent variable structural SVM [210], although the specific details are different.

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

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial \mathbf{A}} &= \tau \mathbf{A} + \sum_{i,m} \eta_{i,m} (\mathbf{w}_m^\top - \mathbf{w}_{y_i}^\top) \frac{\partial \mathbf{f}(\mathbf{v}_i; \mathbf{A})}{\partial \mathbf{A}} \\ &= \tau \mathbf{A} + \sum_{i,m} \eta_{i,m} (\mathbf{w}_m - \mathbf{w}_{y_i}) \mathbf{v}_i^\top \end{aligned} \tag{13}$$

Setting $\frac{\partial \mathcal{L}}{\partial \mathbf{A}} = 0$, the closed-form solution of $\mathbf{A}$ can be obtained as,

$$\begin{aligned} \mathbf{A} &= \frac{1}{\tau} \sum_{i,m} \eta_{i,m} (\mathbf{w}_m - \mathbf{w}_{y_i}) \mathbf{v}_i^\top \\ &= \frac{1}{\tau} \sum_{i,j} (\mathbf{w}_{y_i} - \sum_m \eta_{i,m} \mathbf{w}_m) \mathbf{v}_i^\top, \end{aligned} \tag{14}$$

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 $\sum_m \eta_{i,m} \mathbf{w}_m$.

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

can be further simplified as

$$\mathbf{A} = \frac{1}{\tau} \sum_{\mathbf{v}_i \in \mathrm{SV}} (\mathbf{w}_{y_i} - \sum_m \eta_{i,m} \mathbf{w}_m) \mathbf{v}_i^\top \tag{15}$$

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.

Solving the quadratic programming defined by the dual form of the SVM is time-consuming, especially on a large-scale dataset. But on learning the projection matrix $\mathbf{A}$, a speed-up learning can be employed 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 [149] and deep learning models [150].

Specifically, in iteration t, the algorithm randomly chooses a subset of training samples $\mathcal{D}_t$ to train the model. There is no closed-form update to $\mathbf{A}$ based on this small sample, therefore an approximate gradient step is needed,

$$\mathbf{A}_t = (1 - \alpha_t \tau) \mathbf{A}_{t-1} + \alpha_t \Big\{ \sum_{\mathbf{v}_i \in \mathrm{SV}(\mathcal{D}_t)} \Big(\mathbf{w}_{y_i}^{(t)} - \sum_m \eta_{i,m}^{(t)} \mathbf{w}_m^{(t)} \Big) \mathbf{v}_i^\top \Big\}, \tag{16}$$

where α_t is a learning rate. In iteration t, α_t is updated with $\alpha_t = \frac{1}{t}$. After convergence, the weights $\mathbf{w}$ is obtained by applying the SVM over the entire dataset with the final $\mathbf{A}$. The algorithm is summarized in Algorithm 1. While mini-batch 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.

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, parsing operations are chosen in an online fashion

Algorithm 1 Mini-batch learning algorithm

Input: Training set $\mathcal{D}$, Regularization parameters λ and τ, Number of iteration T, Initialization matrix $\mathbf{A}_0$, and Threshold ε
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 16 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 $\mathbf{w}$

— 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.

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. Each element of $\mathbf{A}_0$ is initialized with a uniform random value in the range $[0, 1]$. For mini-batch learning, the batch size is fixed to be 500 training samples (shift-reduce operations) in each iteration.

Additional features This work mainly considers the distributed representation of each EDU in its parsing decisions. But prior work has shown that other, structural features can provide useful information [90]. Therefore, DPLP can be further augmented 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 1. In computing these features, all tokens are downcased,

Table 1: Additional features for RST parsing

Feature	**Examples**
Words at beginning and end of the EDU	⟨Begin-word-Stack1 = **Example 7.** *but*⟩ ⟨Begin-word-Stack1-Queue1 = **Example 8.** *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 [171].	⟨Head-words-Stack2 = **Example 9.** *working*⟩
Length of EDU in tokens	⟨Len-Stack1-Stack2 = ⟨7, 8⟩⟩
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: Parsing results of different models on the RST-DT test set. The results of TSP and HILDA are reprinted from prior work [90, 66].

Method	Matrix Form	+Features	K	Span	Nuclearity	Relation
Prior work						
1. HILDA [66]				**83.0**	68.4	54.8
2. TSP 1-1 [90]				82.47	68.43	55.73
3. TSP SW [90]				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

and numerical features are not binned. The dependency structure and POS tags are obtained from MaltParser [151].

3.1.3 Evaluation

DPLP is evaluated on the RST Discourse Treebank [23], comparing against state-of-the-art results. A further investigation on the information encoded by the projection function is also presented here.

The RST Discourse Treebank (RST-DT) consists of 385 documents, with 347 for training and 38 for testing in the standard split. As this work focuses on relational discourse parsing, it follows prior work [49, 90], and use gold EDU segmentations. The strongest automated RST segmentation methods currently attain 95% accuracy [2].

In the RST-DT, most nodes have exactly two children, one nucleus and one satellite. For non-binary relations, right branching is used to binarize the tree structure. For multi-nuclear relations, the left EDU is chosen as "head" EDU. The vocabulary $\mathcal{V}$ includes all unigrams after down-casing. No other preprocessing is performed. In total, there are 16,250 unique unigrams in $\mathcal{V}$.

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 [41, 196]. $\mathbf{B}_{nmf}$ can be constructed 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. $\mathbf{B}_{nmf}$ is calculated by 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 [34]. In this case, the product $\mathbf{B}\boldsymbol{v}$ is viewed as a composition of the word embeddings, using the simple *additive* composition model proposed by [144]. This work simply uses word embeddings from [34] 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. The special case $\mathbf{A} = \mathbf{I}$ is also considered as baseline.

Our approach is compared with HILDA [66] and TSP [90]. Joty *et al.*[90] 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, the results are reported on both approaches. All results are based on the same gold standard EDU segmentation. However, the results cannot directly compare with the results of [49], because they do not evaluate on the overall discourse *structure*, but rather treat each relation as an individual classification problem.

To evaluate the parsing performance, this work uses the three standard ways to measure the performance: unlabeled (i.e., hierarchical spans) and labeled (i.e., nuclearity and relation) F-score. This evaluation approach to RST parsing is described by

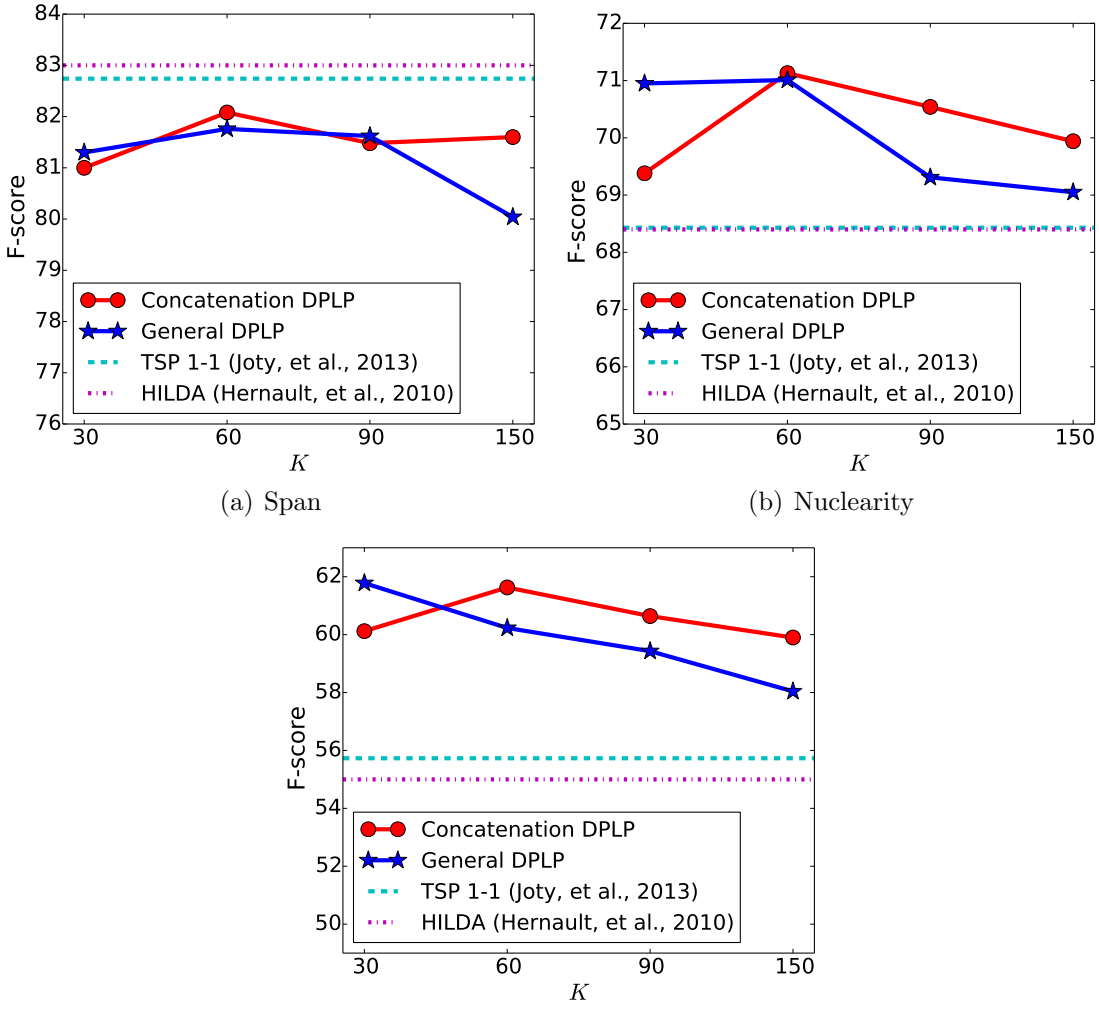
[130]. I implemented the evaluation metrics and make it available with the DPLP system[1]. To compare with prior work of discourse parsing on the RST-DT, this work employed the 18 coarse-grained relations defined in [22].

Table 13 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 1 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 1 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, better results might be obtained 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. As shown in lines 8 and 9, 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

[1] `https://github.com/jiyfeng/DPLP`



(a) Span

(b) Nuclearity

(c) Relation

Figure 14: The performance of our parser over different latent dimension K. Results for DPLP include the additional features from Table 13

projection matrix to model interrelationships between EDUs does not substantially improve performance, and the simpler concatenation construction may be preferred.

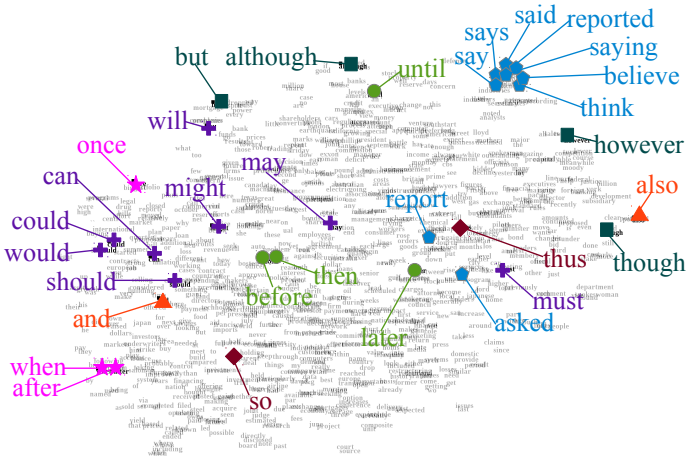
Figure 14 shows how performance changes for different latent dimensions K. At each value of K, grid search was used over a development set to identify the optimal regularizers λ and τ. For the concatenation construction, performance is not overly sensitive to K. For the general form of $\mathbf{A}$, performance decreases with large K. Recall that this construction has nine times as many parameters as the concatenation form; with large values of K, it is likely to overfit.

Why does projection of the surface features improve discourse parsing? To answer this question, I would like to test what information the projection matrix is learning to encoded. First, the projection matrix is token from the concatenation construction and $K = 60$ as an example for case study. Recalling the definition in equation 9, the projection matrix $\mathbf{A}$ will be composed of three identical sub-matrices $\mathbf{B} \in \mathbb{R}^{20 \times V}$. The columns of the $\mathbf{B}$ matrix can be viewed as 20-dimensional descriptors of the words in the vocabulary.

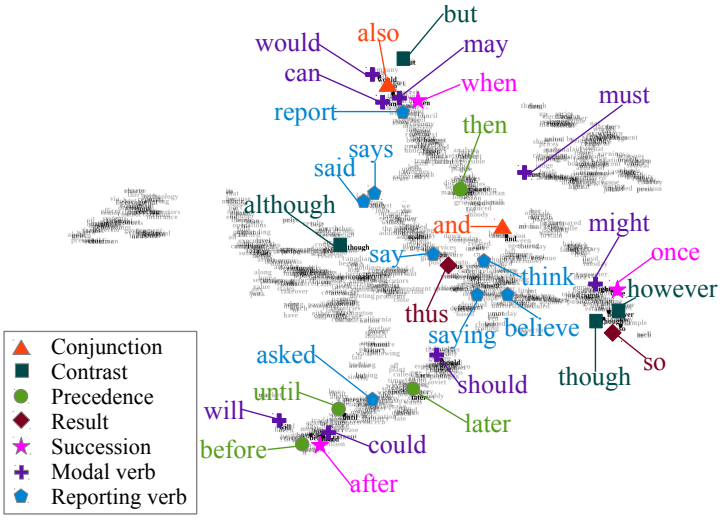
For the purpose of visualization, the dimension of latent representation is reduced from $K = 20$ to 2 dimensions using t-SNE [197]. One further simplification for visualization is to consider only the top 1000 frequent unigrams in the RST-DT training set. For comparison, I also apply t-SNE to the projection matrix $\mathbf{B}_{nmf}$ recovered from non-negative matrix factorization.

Figure 15 highlights words that are related to discourse analysis. Among the top 1000 words, the words are highlighted from 5 major discourse connective categories provided in Appendix B of the PDTB annotation manual [165]: Conjunction, Contrast, Precedence, Result, and Succession. In addition, I also highlighted two verb categories from the top 1000 words: modal verbs and reporting verbs, with their inflections [102].

From the figure, it is clear Dplp has learned a projection matrix that successfully



(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 15: t-SNE Visualization on latent representations of words.

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.

3.2 Distributed Semantic Composition for Implicit Discourse Relation Identification

As demonstrated in the RST-style parsing, most of improvement from representation learning is on relation identification. Considering that relation identification is closely relevant to the semantic understanding of text, this improvement is consistent with the expectation on representation learning. However, the model employed in section 3.1 has a limitation, as discussed on Equation 3 in chapter 1. The distributed representation of a discourse unit is the summarization of distributed representations of words within this unit. This simple composition idea is not enough to capture sufficient semantic information for the relation identification in discourse processing. This section still focuses on idea of representation learning but extends the idea to include more sophisticated ways for sentence-level distributed representation.

As discussed in chapter 1, discourse relations are rooted in semantics [52], which can be difficult to recover from surface level features. Consider again the following example

Example 10. *Bob gave Tina the burger.*
She was hungry.

While a connector like "*because*" seems appropriate here, there is little surface information to signal this relationship, unless the model has managed to learn a bilexical relationship between "*burger*" and "*hungry*". Learning all such relationships

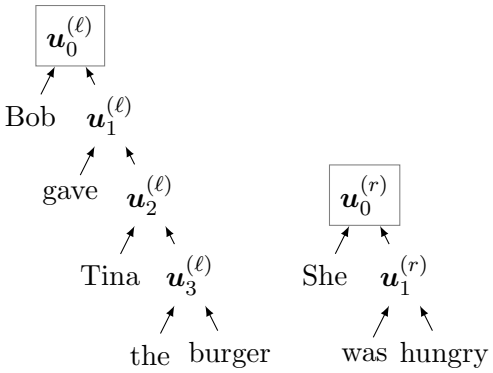


Figure 16: The distributed representations of "*burger*" and "*hungry*" are propagated up the parse tree, clarifying the implicit discourse relation between $\boldsymbol{u}_0^{(\ell)}$ and $\boldsymbol{u}_0^{(r)}$.

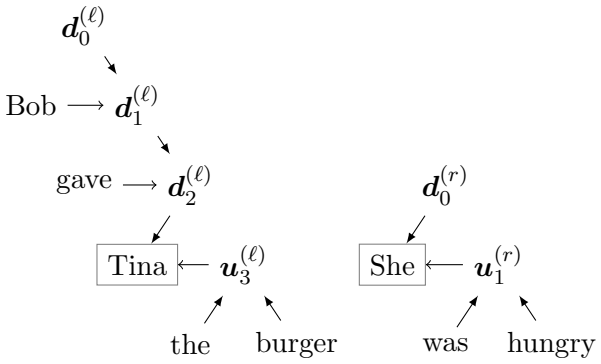


Figure 17: Distributed representations for the coreferent mentions "*Tina*" and "*she*" are computed from the parent and sibling nodes.

from annotated data — including the relationship of "*hungry*" to "*knish*", "*pierogie*", "*pupusa*" etc — would require far more data than can possibly be annotated.

In this section, I present a new way to address this problem based on compositional distributed representation [179, 4]. The meaning of each discourse argument is represented as a vector [195], which is computed through a series of bottom-up compositional operations over the syntactic parse tree. The discourse relation can then be predicted as a bilinear combination of these vector representations. Both the prediction matrix and the compositional operator are trained in a supervised large-margin framework [177], ensuring that the learned compositional operation produces semantic representations that are useful for discourse. Experiments on the PDTB show that this approach outperforms prior work on the classification of implicit discourse

relations, when combined with a small number of surface features.

Despite these positive results, a bottom-up vector-based representations of discourse arguments can be augmented to be insufficient to capture their relations, as discussed in section 1.2. Recall that the tiny modification on the example (10) dramatically changes the discourse relation.

Example 11. *Bob gave Tina the burger.*
He *was hungry.*

After changing the subject of the second sentence to Bob, the connective ""*because*"" no longer seems appropriate; a contrastive connector like "*although*" is preferred. But despite the radical difference in meaning, the bottom-up distributed representation of the second sentence will be almost unchanged: the syntactic structure remains identical, and the words "*he*" and "*she*" have very similar word representations (see Figure 18). If each discourse argument span is reduced into a single vector, built from the elements in the argument itself, a model cannot possibly capture the ways that discourse relations are signaled by entities and their roles [36, 122].

This issue is addressed here by computing vector representations not only for each discourse argument, but also for each coreferent entity mention with an additional composition procedure. These representations are meant to capture the **role** played by the entity in the text, and so they must take the entire span of text into account. The additional procedure computes entity-role representations using a feed-forward compositional model, which combines "upward" and "downward" passes through the syntactic structure, shown in Figure 17. In the example, the downward representations for "*Tina*" and "*she*" are computed from a combination of the parent and sibling nodes in the binarized parse tree. Representations for these coreferent mentions are then combined in a bilinear product, and help to predict the implicit discourse relation.

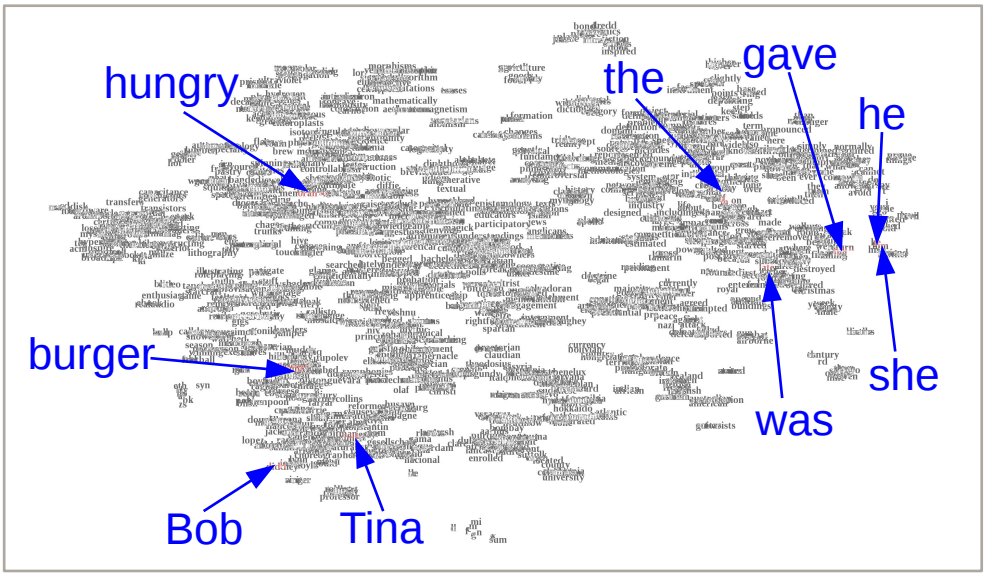


Figure 18: t-SNE visualization [197] of word representations in the PDTB corpus. The pronouns "*she*" and "*he*" are close to each other in the latent space, so it will be nearly impossible for a distributional method to distinguish the meaning of examples.

3.2.1 Entity augmented distributed semantics

Let us define the approach to entity-augmented distributed semantics now. For clarity of exposition, the definition focuses on discourse relations between pairs of sentences. The extension to non-sentence arguments is discussed later in this section.

Distributed representations for discourse arguments are computed in a feed-forward "upward" pass: each non-terminal in the binarized syntactic parse tree has a K-dimensional vector representation that is computed from the representations of its children, bottoming out in pre-trained representations of individual words. This work follows the prior work on Recursive Neural Network (RNN) models proposed by Socher *et al.*[177]. For a given parent node i, let $\ell(i)$ denotes the left child, and $r(i)$ the right child. The representation of node i is given by

$$\boldsymbol{u}_i = \tanh\left(\mathbf{U}[\boldsymbol{u}_{\ell(i)}; \boldsymbol{u}_{r(i)}]\right), \tag{17}$$

where $\tanh(\cdot)$ is the element-wise hyperbolic tangent function [158], and $\mathbf{U} \in \mathbb{R}^{K \times 2K}$

is the upward composition matrix. This compositional procedure is applied recursively from the bottom up, to obtain the sentence-level representation $\boldsymbol{u}_0$. The base case of recursion is the leaf nodes of this composition tree, which are set equal to *pre-trained* word vector representations. For example, in the second sentence of Figure 16, the word representations of "*was*" and "*hungry*" are combined to obtain $\boldsymbol{u}_1^{(r)}$, and then combined with $\boldsymbol{u}_1^{(r)}$ to obtain the word representation of "*she*" $\boldsymbol{u}_0^{(r)}$. Note that the upward pass is feedforward, meaning that there are no cycles and all nodes can be computed in linear time.

As discussed before, a model using only bottom-up representations would find little to distinguish between "*she was hungry*" and "*he was hungry*". It would almost certainly fail to identify the correct discourse relation for at least one of these cases, which requires tracking the roles played by the entities that are coreferent in each pair of sentences. This issue is addressed here with additional composition procedure to represent the semantics of the role played by each coreferent entity in each argument.

In this additional composition, the role of constituent i can be viewed as a combination of information from two neighboring nodes in the parse tree: its parent $\rho(i)$, and its sibling $s(i)$. In other words, a downward pass is used to compute the downward vector $\boldsymbol{d}_i$ from the downward vector of the parent $\boldsymbol{d}_{\rho(i)}$, and the **upward** vector of the sibling $\boldsymbol{u}_{s(i)}$:

$$\boldsymbol{d}_i = \tanh\left(\mathbf{V}[\boldsymbol{d}_{\rho(i)}; \boldsymbol{u}_{s(i)}]\right), \tag{18}$$

where $\mathbf{V} \in \mathbb{R}^{K \times 2K}$ is the downward composition matrix. The base case of this recursive procedure occurs at the root of the parse tree, which is set equal to the upward representation, $\boldsymbol{d}_0 \triangleq \boldsymbol{u}_0$. This procedure is illustrated in Figure 17: for "*Tina*", the parent node is $\boldsymbol{d}_2^{(\ell)}$, and the sibling is $\boldsymbol{u}_3^{(\ell)}$.

This up-down compositional algorithm propagates sentence-level distributed semantics back to entity mentions. The representation of each mention's role in the sentence is based on the corresponding role of the parent node in the parse tree,

and on the internal meaning representation of the sibling node, which is computed by upward composition. Note that this algorithm is designed to maintain the *feed-forward* nature of the neural network, so that we can efficiently compute all nodes without iterating. Each downward node $\boldsymbol{d}_i$ influences only other downward nodes $\boldsymbol{d}_j$ where $j > i$, meaning that the downward pass is feedforward. The upward node is also feedforward: each upward node $\boldsymbol{u}_i$ influences only other upward nodes $\boldsymbol{u}_j$ where $j < i$. Since the upward and downward passes are each feedforward, and the downward nodes do not influence any upward nodes, the combined up-down network is also feedforward. This ensures that all computation of $\boldsymbol{u}_i$ and $\boldsymbol{d}_i$ is linear in the length of the input.

To predict the discourse relation between an argument pair (m, n), the decision function is a sum of bilinear products,

$$\psi(y) = (\boldsymbol{u}_0^{(m)})^\top \mathbf{A}_y \boldsymbol{u}_0^{(n)} + \sum_{i,j \in \mathcal{A}(m,n)} (\boldsymbol{d}_i^{(m)})^\top \mathbf{B}_y \boldsymbol{d}_j^{(n)} + b_y, \tag{19}$$

where $\mathbf{A}_y \in \mathbb{R}^{K \times K}$ and $\mathbf{B}_y \in \mathbb{R}^{K \times K}$ are the classification parameters for relation y. A scalar b_y is used as the bias term for relation y, and $\mathcal{A}(m, n)$ is the set of coreferent entity mentions shared by the argument pair (m, n). The decision value $\psi(y)$ of relation y is therefore based on the upward vectors at the root, $\boldsymbol{u}_0^{(m)}$ and $\boldsymbol{u}_0^{(n)}$, as well as on the downward vectors for each pair of aligned entity mentions. For the cases where there are no coreferent entity mentions between two sentences, $\mathcal{A}(m, n) = \varnothing$, the classification model considers only the upward vectors at the root.

To avoid overfitting, I propose a low-dimensional approximation to each $\mathbf{A}_y$,

$$\mathbf{A}_y = \boldsymbol{a}_{y,1} \boldsymbol{a}_{y,2}^\top + \text{diag}(\boldsymbol{a}_{y,3}). \tag{20}$$

The same approximation is also applied to each $\mathbf{B}_y$, reducing the number of classification parameters from $2 \times \#|\mathcal{Y}| \times K^2$ to $2 \times \#|\mathcal{Y}| \times 3K$.

Prior work has identified a number of useful surface-level features [118], and the classification model can easily be extended to include them. Defining $\boldsymbol{\phi}_{(m,n)}$ as the

vector of surface features extracted from the argument pair (m, n), the corresponding decision function is modified as,

$$\psi(y) = (\boldsymbol{u}_0^{(m)})^\top \mathbf{A}_y \boldsymbol{u}_0^{(n)} + \sum_{i,j \in \mathcal{A}(m,n)} (\boldsymbol{d}_i^{(m)})^\top \mathbf{B}_y \boldsymbol{d}_j^{(n)} + \boldsymbol{\beta}_y^\top \boldsymbol{\phi}_{(m,n)} + b_y, \tag{21}$$

where $\boldsymbol{\beta}_y$ is the classification weight on surface features for relation y. The detail information of these features will be described later.

Connection to the inside-outside algorithm In the inside-outside algorithm for computing marginal probabilities in a probabilistic context-free grammar [106], the inside scores are constructed in a bottom-up fashion, like our upward nodes; the outside score for node i is constructed from a product of the outside score of the parent $\rho(i)$ and the inside score of the sibling $s(i)$, like our downward nodes. The standard inside-outside algorithm sums over all possible parse trees, but since the parse tree is observed in our case, a closer analogy would be to the constrained version of the inside-outside algorithm for latent variable grammars [161]. Cohen *et al.*[32] describe a tensor formulation of the constrained inside-outside algorithm. Similarly, the downward vectors could be computed by a tensor contraction of the parent and sibling vectors [173, 174]. However, this would involve K^3 parameters, rather than the K^2 parameters as in the matrix-vector composition.

3.2.2 Large-margin learning framework

There are two sets of parameters to be learned: the classification parameters $\boldsymbol{\theta}_{class} = \{\mathbf{A}_y, \mathbf{B}_y, \boldsymbol{\beta}_y, b_y\}_{y \in \mathcal{Y}}$, and the composition parameters $\boldsymbol{\theta}_{comp} = \{\mathbf{U}, \mathbf{V}\}$. For leaf nodes, the model uses pre-trained word representations, and do not update them. While my prior work shows that it can be advantageous to retrain word representations for discourse analysis [81], preliminary experiments found that updating the word representations led to serious overfitting in this model.

Following [177], backpropagation with a large margin framework is used to learn

all parameters of the network jointly [57]. Learning is performed using stochastic gradient descent [19]. In the following, the learning procedure is formulated on a single argument pair (m, n) with the gold discourse relation y^*. The objective function for this training example is a regularized hinge loss,

$$\mathcal{L}(\boldsymbol{\theta}) = \sum_{y':y' \neq y^*} \max\Big(0, 1 - \psi(y^*) + \psi(y')\Big) + \lambda||\boldsymbol{\theta}||_2^2 \tag{22}$$

where $\boldsymbol{\theta} = \boldsymbol{\theta}_{class} \cup \boldsymbol{\theta}_{comp}$ is the set of learning parameters. The regularization term $\lambda||\boldsymbol{\theta}||_2^2$ indicates that the squared values of all parameters are penalized by λ; this corresponds to penalizing the squared Frobenius norm for the matrix parameters, and the squared Euclidean norm for the vector parameters.

In Equation 51, $\mathcal{L}(\boldsymbol{\theta}) = 0$, if for every $y' \neq y^*$, $\psi(y^*) - \psi(y') \geq 1$ holds. Otherwise, the loss will be caused by any y', where $y' \neq y^*$ and $\psi(y^*) - \psi(y') < 1$. The gradient for the classification parameters therefore depends on the margin value between gold label and all other labels. Specifically, taking one component of $\mathbf{A}_y$, $\boldsymbol{a}_{y,1}$, as an example, the derivative of the objective for $y = y^*$ is

$$\frac{\partial \mathcal{L}(\boldsymbol{\theta})}{\partial \boldsymbol{a}_{y^*,1}} = -\sum_{y':y' \neq y^*} \delta_{(\psi(y^*)-\psi(y')<1)} \cdot \boldsymbol{u}_0^{(m)}, \tag{23}$$

where $\delta_{(\cdot)}$ is the delta function. The derivative for $y' \neq y^*$ is

$$\frac{\partial \mathcal{L}(\boldsymbol{\theta})}{\partial \boldsymbol{a}_{y',1}} = \delta_{(\psi(y^*)-\psi(y')<1)} \cdot \boldsymbol{u}_0^{(m)} \tag{24}$$

During learning, the updating rule for $\mathbf{A}_y$ is

$$\mathbf{A}_y \leftarrow \mathbf{A}_y - \eta(\frac{\partial \mathcal{L}(\boldsymbol{\theta})}{\partial \mathbf{A}_y} + \lambda \mathbf{A}_y) \tag{25}$$

where η is the learning rate. The gradient information can be obtained in a similar way to update parameters $\{\mathbf{B}_y, \boldsymbol{\beta}_y, b_y\}_{y \in \mathcal{Y}}$.

There are two composition matrices $\mathbf{U}$ and $\mathbf{V}$, corresponding to the upward and

downward composition procedures respectively. Taking the upward composition parameter $\mathbf{U}$ as an example, the derivative of $\mathcal{L}(\boldsymbol{\theta})$ with respect to $\mathbf{U}$ is

$$\frac{\partial \mathcal{L}(\boldsymbol{\theta})}{\partial \mathbf{U}} = \sum_{y':y' \neq y^*} \delta_{(\psi(y^*)-\psi(y')<1)} \cdot \Big(\frac{\partial \psi(y')}{\partial \mathbf{U}} - \frac{\partial \psi(y^*)}{\partial \mathbf{U}} \Big) \tag{26}$$

As with the classification parameters, the derivative depends on the margin between y' and y^*.

For every $y \in \mathcal{Y}$, the unified derivative form exists,

$$\frac{\partial \psi(y)}{\partial \mathbf{U}} = \frac{\partial \psi(y)}{\partial \boldsymbol{u}_0^{(m)}} \frac{\partial \boldsymbol{u}_0^{(m)}}{\partial \mathbf{U}} + \frac{\partial \psi(y)}{\partial \boldsymbol{u}_0^{(n)}} \frac{\partial \boldsymbol{u}_0^{(n)}}{\partial \mathbf{U}} + \sum_{i,j \in \mathcal{A}(m,n)} \frac{\partial \psi(y)}{\partial \boldsymbol{d}_i^{(m)}} \frac{\partial \boldsymbol{d}_i^{(m)}}{\partial \mathbf{U}} + \sum_{i,j \in \mathcal{A}(m,n)} \frac{\partial \psi(y)}{\partial \boldsymbol{d}_j^{(n)}} \frac{\partial \boldsymbol{d}_j^{(n)}}{\partial \mathbf{U}}, \tag{27}$$

The gradient of $\mathbf{U}$ also depends on the gradient of $\psi(y)$ with respect to every downward vector $\boldsymbol{d}$, as shown in the last two terms in Equation 27. This is because the computation of each downward vector $\boldsymbol{d}_i$ includes the upward vector of the sibling node, $\boldsymbol{u}_{s(i)}$, as shown in Equation 18. For an example, see the construction of the downward vectors for "*Tina*" and "*she*" in Figure 17.

The partial derivatives of the decision function in Equation 27 are computed as,

$$\begin{aligned} &\frac{\partial \psi(y)}{\partial \boldsymbol{u}_0^{(m)}} = A_y \boldsymbol{u}_0^{(n)}, \frac{\partial \psi(y)}{\partial \boldsymbol{u}_0^{(n)}} = A_y^\top \boldsymbol{u}_0^{(m)}, \\ &\frac{\partial \psi(y)}{\partial \boldsymbol{d}_i^{(m)}} = B_y \boldsymbol{d}_j^{(n)}, \frac{\partial \psi(y)}{\partial \boldsymbol{d}_i^{(n)}} = B_y^\top \boldsymbol{d}_j^{(m)}, \langle i,j \rangle \in \mathcal{A}. \end{aligned} \tag{28}$$

The partial derivatives of the upward and downward vectors with respect to the upward compositional operator are computed as,

$$\frac{\partial \boldsymbol{u}_i^{(m)}}{\partial \mathbf{U}} = \sum_{\boldsymbol{u}_k^{(m)} \in \mathcal{T}(\boldsymbol{u}_i^{(m)})} \frac{\partial \boldsymbol{u}_i^{(m)}}{\partial \boldsymbol{u}_k^{(m)}} (\boldsymbol{u}_k^{(m)})^\top \tag{29}$$

and

$$\frac{\partial \boldsymbol{d}_i^{(m)}}{\partial \mathbf{U}} = \sum_{\boldsymbol{u}_k^{(m)} \in \mathcal{T}(\boldsymbol{d}_i^{(m)})} \frac{\partial \boldsymbol{d}_i^{(m)}}{\partial \boldsymbol{u}_k^{(m)}} (\boldsymbol{u}_k^{(m)})^\top, \tag{30}$$

where $\mathcal{T}(\boldsymbol{u}_m)$ is the set of all nodes in the upward composition model that help to generate $\boldsymbol{u}_m$. For example, in Figure 16, the set $\mathcal{T}(\boldsymbol{u}_2^{(\ell)})$ includes $\boldsymbol{u}_3^{(\ell)}$ and the word

representations for "*Tina*", "*the*", and "*burger*". The set $\mathcal{T}(\boldsymbol{d}_{m,i})$ includes all the **upward** nodes involved in the downward composition model generating $\boldsymbol{d}_i^{(m)}$. For example, in Figure 17, the set $\mathcal{T}(\boldsymbol{d}_{\text{she}}^{(r)})$ includes $\boldsymbol{u}_1^{(r)}$ and the word representations for "*was*" and "*hungry*".

The derivative of the objective with respect to the downward compositional operator $\mathbf{V}$ is computed in a similar fashion, but it depends only on the downward nodes, $\boldsymbol{d}_i^{(m)}$.

During learning, we used AdaGrad [43] to tune the learning rate in each iteration. To avoid the exploding gradient problem [10], the norm clipping trick proposed in [158] is used in the learning procedure with a fixed norm threshold $\tau = 5.0$.

This model includes three tunable hyperparameters: the latent dimension K for the distributed representation, the regularization parameter λ, and the initial learning rate η. All hyperparameters are tuned by randomly selecting a development set of 20% of the training data. In our experiments, the grid search was performed with $K \in \{20, 30, 40, 50, 60\}$ for the latent dimensionality, $\lambda \in \{0.0002, 0.002, 0.02, 0.2\}$ for the regularization (on each training instance), and $\eta \in \{0.01, 0.03, 0.05, 0.09\}$ for the learning rate. Separate regularizers and learning rates were assigned to the upward composition model, downward composition model, feature model and the classification model with composition vectors.

All the classification parameters are initialized to $\mathbf{0}$. For the composition parameters, the initialization follows [7] and sets $\mathbf{U}$ and $\mathbf{V}$ with uniform random values drawn from the range $[-\sqrt{6/2K}, \sqrt{6/2K}]$. Training on the PDTB takes roughly three hours to converge, on an Intel(R) Xeon(R) CPU 2.20GHz without parallel computing. Convergence is faster if the surface feature weights $\boldsymbol{\beta}$ are trained separately first.

For word representations, a word2vec model [139] was trained on the PDTB corpus. Then, the representation was standardized to zero-mean, unit-variance [112].

Table 3: Proportion of relations with coreferent entities, according to automatic coreference resolution and gold coreference annotation.

Dataset	Annotation	Training (%)	Test (%)
1. PDTB	Automatic	27.4	29.1
2. PDTB∩Onto	Automatic	26.2	32.3
3. PDTB∩Onto	Gold	40.9	49.3

Experiments with pre-trained GloVe word vector representations [160] gave broadly similar results.

In addition, the model also requires that the syntactic structure for each argument is represented as a binary tree. All parses of arguments are obtained by running the Stanford parser [95], and binarizing all resulting parse trees. Argument spans in the Penn Discourse Treebank need not be sentences or syntactic constituents: they can include non-constituent spans, and even discontinuous spans [165]. In all cases, all syntactic subtrees for an argument span was identified, and then unified in a right branching superstructure.

The impact of entity semantics on discourse relation detection is inherently limited by two factors: (1) the frequency with which the arguments of a discourse relation share coreferent entity mentions, and (2) the ability of automated coreference resolution systems to detect these coreferent mentions. To extract entities and their mentions from the PDTB, I ran the Berkeley coreference system [45] on each document. For each argument pair, ignore the non-corefential entity mentions. Line 1 in Table 3 shows the proportion of the instances with shared entities in the PDTB training and test data, as detected by the Berkeley system. As the system does not detect coreferent mentions in more than 70% of the cases, the performance improvements offered by distributed entity semantics are therefore limited. To determine whether this low rate of coreference is an intrinsic property of the data, or whether it is due to the quality of state-of-the-art coreference resolution, it is interesting to also consider

the gold coreference annotations in the OntoNotes corpus [164], a portion of which intersects with the PDTB (597 documents). Lines 2 and 3 of Table 3 give the statistics for automatic and gold coreference on this intersection. These results indicate that with perfect coreference resolution, the applicability of distributed entity semantics would reach 40% of the training set and nearly 50% of the test set. Thus, improvements in coreference resolution can be expected to yield further improvements in the effectiveness of distributed entity semantics for discourse relation detection.

In experiments, the model was supplemented using additional surface features proposed in [118]. These include four categories: word pair features, constituent parse features, dependency parse features, and contextual features. As done in this prior work [118], a criterion based on mutual information to select features in the first three categories, obtaining 500 word pair features, 100 constituent features, and 100 dependency features. In addition, Rutherford and Xue [170] discovered that replacing word pair with their Brown cluster assignments could give further improvements. In the implementation, I used the Brown word clusters provided by [193], in which words from the Reuters Corpus (RCV1) are grouped into 3,200 clusters. The feature selection method of [118] was then used to obtain a set of 600 Brown cluster features.

3.2.3 Evaluation

The model is evaluated on the Penn Discourse Treebank [165], which provides a discourse level annotation over the Wall Street Journal corpus. In the PDTB, each discourse relation is annotated between two argument spans. Identifying the argument spans of discourse relations is a challenging task [119], which we do not attempt here; instead, I used gold argument spans, as in most of the relevant prior work. The evaluation performed here focuses on classifying implicit discourse relations.

There are two main approaches to evaluating implicit discourse relation classification, as discussed in chapter 2. Multiclass classification requires identifying the

discourse relation from all possible choices. This task was first explored in [118], with the focus on second-level discourse relations. More recent work has emphasized binary classification, where the goal is to build and evaluate separate "one-versus-all" classifiers for each discourse relation [162, 156, 13].

3.2.3.1 Multiclass classification

The training and test set construction for the multiclass classification follows [118] with a few changes:

- Sections 2-20 of the PDTB was used as a training set, sections 0-1 as a development set for parameter tuning, and sections 21-22 for testing.
- Five relation types have a combined total of only nine instances in the training set, and are therefore excluded by [118]: CONDITION, PRAGMATIC CONDITION, PRAGMATIC CONTRAST, PRAGMATIC CONCESSION and EXCEPTION. None of these relations appear in the test or development data. I tried training with and without these relation types in the training data, and found no difference in the overall results.
- In the main multiclass experiment, I considered only the problem of distinguishing between implicit relations. I also performed an additional experiment that distinguishes implicit relations from entity-based coherence relations, labeled ENTREL.
- Roughly 2% of the implicit relations in the PDTB are annotated with more than one type. During training, each argument pair that is annotated with two relation types is considered as two training instances, each with one relation type. During testing, if the classifier assigns either of the two types, it is considered to be correct.

The following baseline and competitive systems were adopted for comparison

Most common class The most common class is CAUSE, accounting for 26.03% of the implicit discourse relations in the PDTB test set.

Additive word representations [16] show that simply adding word vectors can perform surprisingly well at assessing the meaning of short phrases. In this baseline, each argument was represented as a sum of its word representations, and estimated with a bilinear prediction matrix.

Lin *et al.*, (2009) [118] This is the then-best published accuracy on multiclass classification of second-level implicit discourse relations is from [118], which applies feature selection to obtain a set of lexical and syntactic features over the arguments.

Surface features + Brown clusters To get a more precise comparison, I reimplemented the system of [118]. The major differences are (1) I used the online learning framework as proposed in the model part, rather than batch classification, and (2) I also included the Brown cluster features described before and originally proposed by [170].

Compositional Finally, the results for the compositional methods. Since it is a **dis**tributional **co**mpositional approach to **disco**urse relations, it is named as DISCO2.

Table 4 presents results for multiclass identification of second-level PDTB relations. As shown in lines 7 and 8, DISCO2 outperforms both baseline systems and the prior state-of-the-art (line 3). The strongest performance is obtained by including the entity distributed semantics, with a 4.4% improvement over the accuracy reported by [118] ($p < .05$ by a binomial test). A significant improvement is obtained over the Surface Feature + Brown Cluster model. Because I have the implementation of this system, therefore can use the sign test for statistical significance. Again, DISCO2 is

Table 4: Experimental results on multiclass classification of level-2 discourse relations. The results of Lin *et al.* (2009) [118] are shown in line 3. We reimplemented this system and added the Brown cluster features from Rutherford and Xue (2014) [170], with results shown in line 4.

Model	+Entity semantics	+Surface features	K	Accuracy(%)
Baseline models				
1. Most common class				26.03
2. Additive word representations			50	28.73
Prior work				
3. Lin *et al.* (2010) [118]		✓		40.2
Our work				
4. Surface features + Brown clusters		✓		40.66
5. DISCO2			50	36.98
6. DISCO2	✓		50	37.63
7. DISCO2		✓	50	43.75*
8. DISCO2	✓	✓	50	44.59*

* signficantly better than lines 3 and 4 with $p < 0.05$

significantly better the baseline system ($p < .05$). However, the surface features remain important, as the performance of DISCO2 is substantially worse when only the distributed representation is included. The latent dimension K is chosen from a development set, as shown in Figure 19.

Another question is whether it is possible to identify entity-based coherence, annotated in the PDTB as ENTREL, which is when a shared entity is the only meaningful relation that holds between two sentences [165]. To answer this question, I added ENTREL to the set of possible relations, and performed an additional evaluation. Since this setting has not previously been considered, it cannot be evaluated against published results; instead, I retrained and evaluated the following models:

- the surface feature baseline with Brown clusters, corresponding to line 4 of Table 4;
- DISCO2 with surface features but without entity semantics, corresponding to line 7 of Table 4;
- DISCO2 with surface features and entity semantics, corresponding to line 8 of

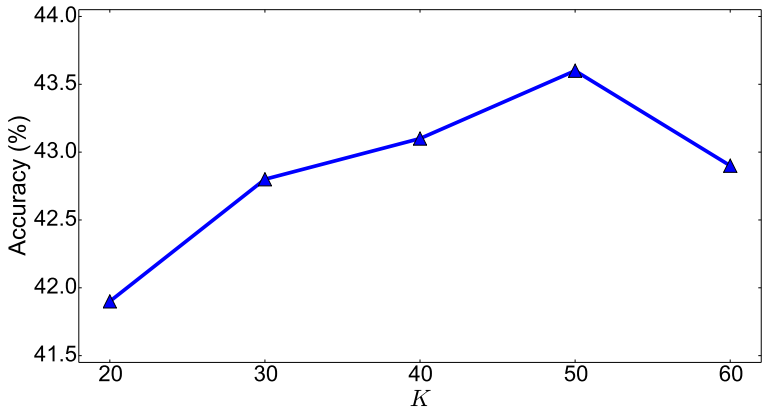


Figure 19: The performance of DISCO2 (full model), over different latent dimensions K.

Table 4.

As before, all parameters are tuned on a development set. In this evaluation, a larger improvements is obtained from DISCO2: the full model (with entity semantics) gives 47.27% accuracy, as compared to 44.96% without entity semantics; the result for the surface feature baseline is 41.48%.

The contribution of entity semantics is shown in Table 4 by the accuracy differences between lines 5 and 6, and between lines 7 and 8. On the subset of relations in which the arguments share at least one coreferent entity, the difference is substantially larger: the accuracy of DISCO2 is 45.7% with entity mention semantics, and 43.1% without. Considering that only 29.1% of the relations in the PDTB test set include shared entities, it therefore seems likely that a more sensitive coreference system could yield further improvements for the entity-semantics model. Indeed, gold coreference annotation on the intersection between the PDTB and the OntoNotes corpus shows that 40-50% of discourse relations involve coreferent entities (Table 3). Evaluating on just this intersection, the inclusion of entity semantics yields an improvement in accuracy from 37.5% to 39.1%. Thus, while the overall improvements offered by entity mention semantics are relatively small, this is due in part to the poor recall of the state-of-the-art coreference resolution system; if coreference improved, the impact of

the entity mention semantics would increase correspondingly.

A question about whether it was necessary to have the correct coreference alignment, or whether similar improvements could be obtained by computing bilinear products between all pairs of noun phrases in the two discourse arguments. In fact, this strategy of aligning all entity mentions resulted in a decrease in accuracy, from 44.59 to 42.14%. This is below the performance of DISCO2 without entity semantics.

Examples The following examples help highlight how entity semantics can improve the accuracy of discourse relation classification. In example 12, the entity-augmented model correctly identifies the relation as RESTATEMENT, due in part to the detected coreference between "*The Wall Street Journal*" and "*the Journal*": in both arguments, the entity experiences a drop in profits. Without this information, DISCO2 incorrectly labels this relation as CAUSE. In example 13, the entity-augmented model correctly identifies the relation as CONTRAST, which is reasonable given the very different role of the shared entity "*Mr. Greenberg*" in the two arguments; without entity semantics, it is classified as CONJUNCTION. Example 14 is more complex because it involves two entities, but again, the CONTRAST relation is correctly detected, in part because of the differing experiences of the two entities in the two arguments; without entity semantics, this example is again incorrectly classified as CONJUNCTION.

Example 12.

Arg1: *The drop in profit reflected, in part, continued softness in financial advertising at The Wall Street Journal and Barron's magazine.*

Arg2: *Ad linage at the Journal fell 6.1% in the third quarter.*

Example 13.

Arg1: *Mr. Greenberg got out just before the 1987 crash and, to his regret, never went back even as the market soared.*

Arg2: *This time he's ready to buy in "when the panic wears off."*

Example 14.

Arg1: *Half of them_1 are really scared and want to sell but I_2'm trying to talk them out of it.*

Arg2: *If they_1 all were bullish, I_2'd really be upset.*

3.2.3.2 Binary classification

Much of the recent work in PDTB relation detection has focused on binary classification, building and evaluating separate one-versus-all classifiers for each relation type [162, 156, 13]. This work has focused on recognition of the four *first*-level relations, grouping EntRel with the Expansion relation. We follow this evaluation approach as closely as possible, using sections 2-20 of the PDTB as a training set, sections 0-1 as a development set for parameter tuning, and sections 21-22 for testing.

The evaluation was performed with the full system. However, instead of a single multiclass classifier for all four relations, four binary classifiers were trained, one for each first-level discourse relation. The hyperparameters K, λ, η were optimized separately for each classifier, by performing a grid search to optimize the F-measure on the development data. Following the work in [162], a balanced training set was constructed by resampling training instances in each class until the number of positive

Table 5: Evaluation on the first-level discourse relation identification. The results of the competitive systems are reprinted.

	Comparison		Contingency		Expansion		Temporal	
	F1	Acc	F1	Acc	F1	Acc	F1	Acc
Competitive systems								
1. Pitler *et al.* (2009) [162]	21.96	56.59	47.13	67.30	76.42	63.62	16.76	63.49
2. Zhou *et al.* (2010) [213]	31.79	58.22	47.16	48.96	70.11	54.54	20.30	55.48
3. Park and Cardie (2012) [156]	31.32	**74.66**	49.82	72.09	79.22	69.14	26.57	79.32
4. Biran and McKeown (2013) [13]	25.40	63.36	46.94	68.09	75.87	62.84	20.23	68.35
Our work								
5. disco2	**35.93**	70.27	**52.78**	**76.95**	**80.02**	**69.80**	**27.63**	**87.11**

and negative instances are equal.

The performance is compared against the published results from several competitive systems, including

Pitler *et al.*(2009) [162] present a classification model using linguistically-informed features, such as polarity tags and Levin verb classes.

Zhou *et al.*(2010) [213] predict discourse connective words, and then use these predicted connectives as features in a downstream model to predict relations.

Park and Cardie (2012) [156] show that the performance on each relation can be improved by selecting a locally-optimal feature set.

Biran and McKeown (2013) [13] reweight word pair features using distributional statistics from the Gigaword corpus, obtaining denser aggregated score features.

Table 5 presents the performance of the disco2 model and the published results of competitive systems. disco2 achieves the best results on most metrics, achieving F-measure improvements of 4.14% on Comparison, 2.96% on Contingency, 0.8% on Expansion, and 1.06% on Temporal. These results are obtained without performing per-relation feature selection, as in prior work. While computing significance over F-measures is challenging, I computed statistical significance on the accuracy results by using the binomial test. disco2 is significantly more accurate than all other

systems on the CONTINGENCY and TEMPORAL relations $p \ll .001$, not significantly more accurate on the EXPANSION relation, and significantly less accurate than the [156] system on the COMPARISON relation at $p \ll .001$.

3.3 Discussion

Discourse relations are determined by the meaning of their arguments, and progress on discourse parsing therefore requires computing representations of the argument semantics. This chapter presents a set of representation learning models with supervision information. The work in section 3.1 demonstrates the power of word representation learning with a simple composition function. Given the limited annotated documents in the RST Treebank, the simple composition function is a trade-off between the availability of data and complexity of models. By adding some additional surface features, the RST parser is further improved the parsing performance, especially relation identification, on the RST Treebank. The low dimensional representation also captures basic intuitions about discourse connectives and verbs, as shown in 15(a).

Section 3.2 further presents a compositional method for inducing distributed representations not only of discourse arguments, but also of the entities that thread through the discourse. In this approach, semantic composition is applied up the syntactic parse tree to induce the argument-level representation, and then down the parse tree to induce representations of entity spans. Discourse arguments can then be compared in terms of their overall distributed representation, as well as by the representations of coreferent entity mentions. This enables the compositional operators to be learned by backpropagation from discourse annotations. In combination with traditional surface features, this approach outperforms previous work on classification of implicit discourse relations in the Penn Discourse Treebank. While the entity mention representations offer only a small improvement in overall performance,

this is limited by the recall of the coreference resolution system: when evaluated on argument pairs for which coreference is detected, the raw improvement from entity semantics is more than 2%.

CHAPTER IV

DISCOURSE-DRIVEN LANGUAGE MODELING

Work described in this chapter was undertaken in collaboration with Trevor Cohn, Chris Dyer, Jacob Eisenstein, Gholamreza Haffari and Lingpeng Kong, published at ICLR 2016 (Workshop track) [79] and NAACL 2016 [83].

Statistical language models are essential components of natural language processing systems, such as machine translation [99], automatic speech recognition [91], text generation [181] and information retrieval [126]. Language models estimate the probability of a word for a given context. In conventional language models, context is represented by n-grams, so these models condition on a fixed number of preceding words. Recurrent Neural Network Language Models (RNNLMs) [137] use a dense vector representation to summarize context across all preceding words within the same sentence. Context operates on multiple levels of detail: on the syntactic level, a word's immediate neighbors are most predictive; on the level of discourse and topic, all words in the document lend contextual information.

Recent research has developed a variety of ways to incorporate document-level contextual information. For example, topic information extracted from the entire document is used [140, 110] to help predict words in each sentence; the bag-of-words representation of the previous sentence is also proposed to construct contextual information with a separate model [117]; a similar bag-of-words context is used in [202] to integrate contextual information into a LSTM for generating the current sentence. These models are all hybrid architectures in that they are recurrent at the sentence level, but use a different architecture to summarize the context outside the sentence.

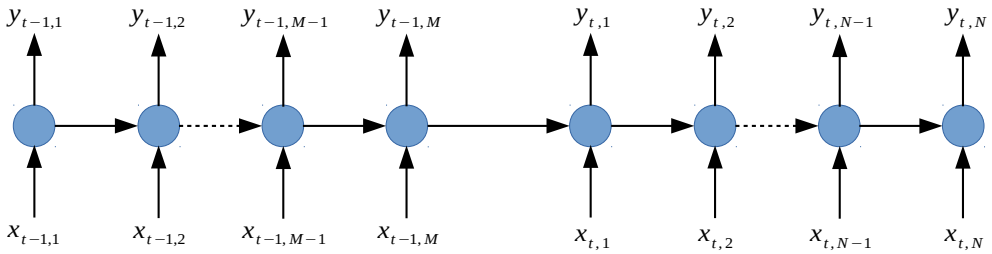


Figure 20: A fragment of document-level recurrent neural network language model (DRNNLM). It is also an extension of sentence-level RNNLM to the document level by ignoring sentence boundaries.

The simplest such model would be to train a single RNN, ignoring sentence boundaries: as shown in Figure 20, the last hidden state from the previous sentence $t-1$ is used to initialize the first hidden state in sentence t. In such an architecture, the length of the RNN is equal to the number of tokens in the document; in typical genres such as news texts, this means training RNNs from sequences of several hundred tokens, which introduces two problems:

Information decay In a sentence with thirty tokens (not unusual in news text), the contextual information from the previous sentence must be propagated through the recurrent dynamics thirty times before it can reach the last token of the current sentence. Meaningful document-level information is unlikely to survive such a long pipeline.

Learning It is notoriously difficult to train recurrent architectures that involve many time steps [10]. In the case of an RNN trained on an entire document, back-propagation would have to run over hundreds of steps, posing severe numerical challenges.

In this chapter, I explore recurrent architectures for combining discourse information in language modeling. section 4.2 presents a set of document context language models (DCLMs) with contextual information summarized from preceding text in document. Each DCLM in section 4.2 represent a possible way of utilizing contextual

information for word generation. Based on the success of DCLMs on language modeling, a further extension of one DCLM (coDCLM) is to include discourse relations between adjacent sentences, named DrLM. section 4.3 includes the detail discussion on DrLM, and demonstrates its capacity on discourse driven language models and discourse relation prediction.

4.1 Prior Work on Language Modeling

Neural language models (NLMs) learn the distributed representations of words together with the probability function of word sequences. In the NLM proposed in [9], a feed-forward neural network with a single hidden layer was used to calculate the language model probabilities. One limitation of this model is only fixed-length context can be used. Recurrent neural network language models (RNNLMs) avoid this problem by recurrently updating a hidden state [137], thus enabling them to condition on arbitrarily long histories. In this work, a sentence-level RNNLM is extend to include more context with a recurrent architecture, by allowing multiple pathways for historical information to affect the current word. A comprehensive review of recurrent neural networks language models is offered in [39].

Conventional language models, including the models with recurrent structures [137], limit the context scope within a sentence. This ignores potentially important information from preceding text, for example, the previous sentence. Targeting speech recognition, where contextual information may be especially important, [140] introduce the *topic-conditioned* Rnnlm, which incorporates a separately-trained latent Dirichlet allocation topic model to capture the broad themes of the preceding text. This chapter focuses on discriminatively-trained end-to-end models.

Lin *et al.*[117] recently introduced a document-level language model, called hierarchical recurrent neural network language model (hRnnlm). There are two channels of information: a RNN for modeling words in a sentence, and another recurrent model

for modeling sentences, based on a bag-of-words representation of each sentence. Contemporaneously to our model, Want and Cho [202] construct a bag-of-words representation of previous sentences, which they then insert into a sentence-level LSTM. The modeling approach proposed in language modeling with contextual information is more unified and compact — employing a single recurrent neural network architecture, but with multiple channels for information to feed forward into the prediction of each word.

Except the document-level language modeling, there are also some approaches to directly model document content. Li *et al.*[114] propose to use a convolution kernel to summarize sentence-level representations for modeling a document. The model is for coherence evaluation, in which the parameters are learned via supervised training. Related convolutional architectures for document modeling are considered in [40, 190]. Encoder-decoder architectures provide an alternative perspective, compressing all the information in a sequence into a single vector, and then attempting to decode the target information from this vector; while this idea has notably applied in machine translation [29], it can also be employed for coherence modeling [116]. The hierarchical sequence-to-sequence model of [116] conditions the start word of each sentence on contextual information provided by the encoder, but does not apply this idea to language modeling. Different from the models with hierarchical structures, paragraph vector [110] encodes a document to a numeric vector by discarding document structure and only retaining topic information.

4.2 Language modeling with document context

This section presents a set of language models to integrate contextual information from previous sentences. Each of them has either some practical or theoretical merits.

4.2.1 Recurrent Neural Network Language Models

To describe the document context language models, I start with a recurrent neural network language model (RNNLM) to explain some necessary terms. Given a sentence $\{\boldsymbol{x}_n\}_{n=1}^N$, a recurrent neural network language model is defined as

$$\begin{aligned} \boldsymbol{h}_n =& \boldsymbol{g}\left(\boldsymbol{h}_{n-1}, \boldsymbol{x}_n\right) \\ \boldsymbol{y}_n =& \text{softmax}\left(\mathbf{W}_o \boldsymbol{h}_n + \boldsymbol{b}\right), \end{aligned} \tag{31}$$

where $\boldsymbol{x}_n \in \mathbb{R}^K$ is the distributed representation of the n-th word, $\boldsymbol{h}_n \in \mathbb{R}^H$ is the corresponding hidden state computed from the word representation and the previous hidden state $\boldsymbol{h}_{n-1}$, and $\boldsymbol{b}$ is the bias term. K and H are the input and hidden dimension respectively. As in the original RNNLM [137], $\boldsymbol{y}_n$ is a prediction of the $(n+1)$-th word in the sequence.

The transition function $\boldsymbol{g}(\cdot)$ could be any nonlinear function used in neural networks, such as the element-wise sigmoid function, or more complex recurrent functions such as the LSTM [73] or GRU [30]. This work uses LSTM, as it consistently gives the best performance in evaluation. By stacking two LSTM together, the new model is able to obtain a even more powerful transition function, called multi-layer LSTM [187]. In a multi-layer LSTM, the hidden state from a lower-layer LSTM cell is used as the input to the upper-layer, and the hidden state from the final-layer is used for prediction. In the following description, the number of layers is fixed as two.

4.2.2 Document Context Language Models

The underlying assumption of this work is that contextual information from previous sentences needs to be able to "short-circuit" the standard RNN, so as to more directly impact the generation of words across longer spans of text. First, the contextual information is represented by the final hidden representation from the previous sentence $t-1$,

$$\boldsymbol{c}_{t-1} = \boldsymbol{h}_{t-1,M} \tag{32}$$

where M is the length of sentence $t-1$. Then additional paths are created for this information to impact each hidden representation in the current sentence t. Let $\boldsymbol{x}_{t,n}$ to be the word representation of the n-th word in the t-th sentence, then

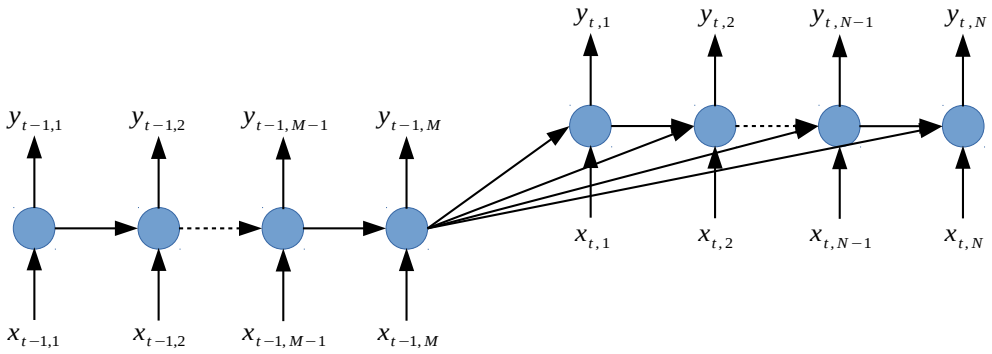
$$\boldsymbol{h}_{t,n} = \boldsymbol{g}_{\boldsymbol{\theta}}\left(\boldsymbol{h}_{t,n-1}, \boldsymbol{s}\left(\boldsymbol{x}_{t,n}, \boldsymbol{c}_{t-1}\right)\right) \tag{33}$$

where $\boldsymbol{g}_{\boldsymbol{\theta}}(\cdot)$ is the activation function parameterized by θ and $\boldsymbol{s}(\cdot)$ is a function that combines the context vector with the input $\boldsymbol{x}_{t,n}$ for the hidden state. The function $\boldsymbol{s}()$ is simply realized by concatenating two representations $\boldsymbol{x}_{t,n}$ and $\boldsymbol{c}_{t-1}$,

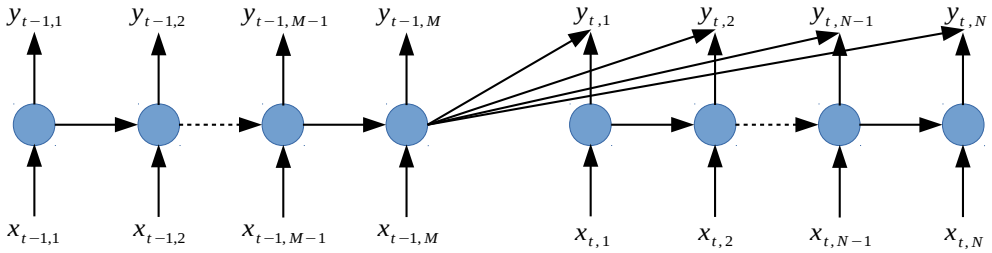
$$\boldsymbol{s}\left(\boldsymbol{x}_{t,n}, \boldsymbol{c}_{t-1}\right) = [\boldsymbol{x}_{t,n}, \boldsymbol{c}_{t-1}]. \tag{34}$$

More sophisticated form can be considered later. The emission probability for $\boldsymbol{y}_{t,n}$ is then computed from $\boldsymbol{h}_{t,n}$ as in the standard RNNLM (Equation 31). The underlying assumption of this model is that contextual information should impact the generation of each word in the current sentence. The model therefore introduces computational "short-circuits" for cross-sentence information, as illustrated in Figure 21(a). Because information flows from one hidden vector to another, this model is called the **context-to-context Document Context Language Model**, abbreviated as ccDCLM.

With this specific architecture, the number of parameters is $H(16H + 3K + 6) + V(H + K + 1)$, where H is the size of the hidden representation, K is the size of the word representation, and V is the vocabulary size. The constant factors come with the weight matrices within a two-layer LSTM unit. This is in the same complexity class as the standard RNNLM. Special handling is necessary for the first sentence of the document. In this case, a *dummy* contextual representation $\boldsymbol{c}_0$ is introduced as a START symbol for a document, just like the START token in sentence-level language modeling. This is another parameter to be learned jointly with the other parameters in this model. The training procedure of ccDCLM is similar to a conventional RNNLM: the model moves from left to right through the document and compute a



(a) ccDCLM



(b) coDCLM

Figure 21: Context-to-context and context-to-output DCLMs

softmax loss on each output $\boldsymbol{y}_{t,n}$. The loss then is backpropagated through the entire sequences.

Rather than incorporated into the recurrent definition of the hidden state, the document context can also be added directly to the output, as illustrated in Figure 21(b). Let $\boldsymbol{h}_{t,n}$ be the hidden state from a conventional RNNLM of sentence t,

$$\boldsymbol{h}_{t,n} = \boldsymbol{g}_\theta \left(\boldsymbol{h}_{t,n-1}, \boldsymbol{x}_{t,n} \right). \tag{35}$$

Then, the context vector $\boldsymbol{c}_{t-1}$ is directly used in the output layer as

$$\boldsymbol{y}_{t,n} \sim \text{softmax} \left(\mathbf{W}_h \boldsymbol{h}_{t,n} + \mathbf{W}_c \boldsymbol{c}_{t-1} + \boldsymbol{b} \right) \tag{36}$$

where $\boldsymbol{c}_{t-1}$ is defined in Equation 32. Because the document context impacts the output directly, this model is named as the **context-to-output DCLM** (coDCLM). The modification on the model architecture from ccDCLM to coDCLM leads to a notable change on the number of parameters. The total number of parameters of

coDCLM is $H(13H + 3K + 6) + V(2H + K + 1)$. The difference of the parameter numbers between these coDCLM and ccDCLM is $VH - 3H^2$. Recall that V is the vocabulary size and H is the size of latent representation, in most cases $V \geq 10^4$ and $H \approx 10^2$. Therefore $V \gg H$ in all reasonable cases, and coDCLM includes more parameters than ccDCLM in general.

While the coDCLM has more parameters that must be learned, it has a potentially important computational advantage. By shifting $\boldsymbol{c}_{t-1}$ from hidden layer to output layer, the relationship of any two hidden vectors $\boldsymbol{h}_t$ and $\boldsymbol{h}_{t'}$ from different sentences is decoupled, so that each can be computed in isolation. In a guided language generation scenario such as machine translation or speech recognition — the most common use case of neural language models — this means that decoding decisions are only *pairwise dependent* across sentences. This is in contrast with the ccDCLM, where the tying between each $\boldsymbol{h}_t$ and $\boldsymbol{h}_{t+1}$ means that decoding decisions are *jointly dependent* across the entire document. This joint dependence may have important advantages, as it propagates contextual information further across the document; the ccDCLM and coDCLM thereby offer two points on a tradeoff between accuracy and decoding complexity.

One potential shortcoming of ccDCLM and coDCLM is the limited capacity of the context vector, $\boldsymbol{c}_{t-1}$, which is a fixed dimensional representation of the context. While this might suffice for short sentences, as sentences grow longer, the amount of information needing to be carried forward will also grow, and therefore a fixed size embedding may be insufficient. For this reason, it may be necessary to consider an *attentional* mechanism, based on conditional language models for translation [3, 187] which allows for a dynamic capacity representation of the context.

Central to the attentional mechanism is the context representation, which is defined separately for each word position in the output sentence,

$$\boldsymbol{c}_{t-1,n} = \sum_{m=1}^{M} \alpha_{n,m} \boldsymbol{h}_{t-1,m} \tag{37}$$

$$\boldsymbol{\alpha}_n = \text{softmax}\left(\boldsymbol{a}_n\right) \tag{38}$$

$$a_{n,m} = \boldsymbol{w}_a^\top \tanh\left(\mathbf{W}_{a1}\boldsymbol{h}_{t,n} + \mathbf{W}_{a2}\boldsymbol{h}_{t-1,m}\right) \tag{39}$$

where $\boldsymbol{c}_{t-1,n}$ is formulated as a weighted linear combination of all the hidden states in the previous sentence, with weights α constrained to lie on the simplex using the softmax transformation. Each weight $\alpha_{n,m}$ encodes the importance of the context at position m for generating the current word at n, defined as a neural network with a hidden layer and a single scalar output. Consequently each position in the generated output can "attend" to different elements of the context sentence, which would arguably be useful to shift the focus to make best use of the context vector during generation.

The revised definition of the context in Equation 37 requires some minor changes in the generating components. This is included as an additional input to both the recurrent function (similar to ccDCLM), and output generating function (akin to coDCLM), as follows

$$\boldsymbol{h}_{t,n} = \boldsymbol{g}_{\boldsymbol{\theta}}\left(\boldsymbol{h}_{t,n-1}, \left[\boldsymbol{c}_{t-1,n}^\top, \boldsymbol{x}_{t,n}^\top\right]^\top\right) \tag{40}$$

$$\boldsymbol{y}_{t,n} \sim \text{softmax}\left(\mathbf{W}_o \tanh\left(\mathbf{W}_h\boldsymbol{h}_{t,n} + \mathbf{W}_c\boldsymbol{c}_{t-1,n} + \boldsymbol{b}\right)\right) \tag{41}$$

where the output uses a single hidden layer network to merge the local state and context, before expanding the dimensionality to the size of the output vocabulary, using $\mathbf{W}_o$. The extended model is named as **attentional DCLM** (aDCLM).

4.2.3 Evaluation

The models were evaluated with perplexity and document-level coherence assessment. The first data set used for evaluation is the Penn Treebank (PTB) corpus [133],

Table 6: Basic statistics of the Penn Treebank (PTB) and North American News Text (NANT) data sets

			Average Document Length	
		# Documents	# Tokens	# Sentences
PTB	Training	2,000	502	21
	Development	155	516	22
	Test	155	577	24
NANT	Training	26,462	783	32
	Development	148	799	33
	Test	2,753	778	32

which is a standard data set used for evaluating language models [137]. I used the standard split proposed by Mikolov *et al.* [137]: sections 0-20 for training, 21-22 for development, and 23-24 for test. For preprocessing, top 10,000 words were kept to construct the vocabulary, and lower frequency words were replaced with the special token UNKNOWN. The vocabulary also includes two special tokens START and STOP to indicate the beginning and end of a sentence. In total, the vocabulary size is $10,003$.

To investigate the capacity of modeling documents with larger context, a subset of the North American News Text (NANT) corpus [134] was also used to construct another evaluation data set. As shown in Table 6, the average length of the training documents is more than 30 sentences. The same preprocessing procedure was used on this dataset, and the top 15,000 words from the training set were kept in the vocabulary. Some basic statistics of both data sets are listed in Table 6.

All models are implemented in the CNN package (`https://github.com/clab/cnn`) with a two-layer LSTM. which is available online at `https://github.com/jiyfeng/dclm`.

All parameters are initialized with random values drawn from the range

$$[-\sqrt{6/(d_1+d_2)}, \sqrt{6/(d_1+d_2)}],$$

where d_1 and d_2 are the input and output dimensions of the parameter matrix respectively, as suggested by [54]. Online learning was performed using AdaGrad [43] with the initial learning $\lambda = 0.1$. To avoid the exploding gradient problem, the norm clipping trick proposed in [158] was used with a fixed norm threshold as $\tau = 5.0$. All models include two tunable hyper-parameters: the dimension of word representation K and the hidden dimension of LSTM unit H. I considered the values $\{32, 48, 64, 96, 128, 256\}$ for both K and H. The best combination of K and H for each model was selected by the development sets via grid search. In all experiments, the hidden dimension of the attentional component in ADCLM is fixed as 48.

As shown in Table 6, the average length of documents is more than 500 tokens, with extreme cases having over 1,000 tokens. In practice, training on long documents leads to a very slow convergence rate. Therefore, each long document was segmented into several non-overlapping shorter documents with at most L sentences and the original sentence order. The value of L used in most experiments is 5, although a comparison with $L = 10$ is included in evaluation.

All three DCLM-style models (CCDCLM, CODCLM, ADCLM) are compared with the following competitive alternatives:

Recurrent neural network language model (RNNLM) The model is trained on individual sentences without any contextual information [137]. The comparison between DCLMs and this baseline system highlights the contribution of contextual information.

RNNLM w/o sentence boundary (DRNNLM) This is a straightforward extension of sentence-level RNNLM to document-level, as illustrated in Figure 20. It can also be viewed a conventional RNNLM without considering sentence boundaries. The difference between RNNLM and DRNNLM is that DRNNLM is able to consider (a limited amount of) extra-sentential context.

Table 7: Perplexities of the Penn Treebank (PTB) and North American News Text (NANT) data sets.

	PTB		NANT	
Model	Dev	Test	Dev	Test
Baselines				
1. RNNLM [137]	69.24	71.88	109.48	194.43
2. RNNLM w/o sentence boundary (dRnnlm)	65.27	69.37	101.42	181.62
3. Hierarchical RNNLM (hRnnlm) [117]	66.32	70.62	103.90	175.92
Our models				
4. Attentional DCLM (aDCLM)	64.31	68.32	96.47	**170.99**
5. Context-to-output DCLM (coDCLM)	64.37	68.49	**95.10**	173.52
6. Context-to-context DCLM (ccDCLM)	**62.34**	**66.42**	96.77	172.88

Hierarchical RNNLM (hRnnlm) To compare with other context model, the architecture of HRNNLM [117] is also adopted as another baseline system, and reimplemented with several modifications for a fair comparison. First, the sigmoid recurrence function in the original implementation was replaced with a long short-term memory (LSTM) as in DCLMs. Furthermore, instead of using pretrained word embedding, the word representation in this new implementation was also updated during training. Finally, the model was jointly trained on both sentence-level and document-level, which is also different from the original implementation. These changes resulted in substantial improvements over the original version of the hRnnlm; they allow the comparison to isolate the most substantive difference between the DCLM and this modeling approach — how contextual information is identified and exploited.

4.2.3.1 Language modeling

Table 7 presents the results on language modeling perplexity. The best perplexities are given by the context-to-context DCLM on the PTB data set (line 6 in Table 7), and attentional DCLM on the NANT data set (line 4 in Table 7). All DCLM-based models achieve better perplexity than the prior work. While the improvements on the PTB dataset are small in an absolute sense, they consistently point to the value of

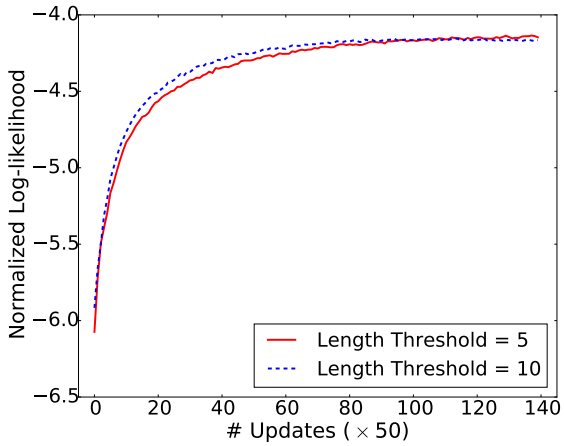


Figure 22: Effect of length thresholds on predictive log-likelihood on the PDTB development set.

including multi-level context information in language modeling. The value of context information is further verified by the model performance on the NANT dataset.

Of interest is the failure of the attentional DCLM to improve performance; while this model is considerably more expressive than the coDCLM and ccDCLM, it is considerably more complex to learn. Manual investigation showed that the attentional state was fairly static in most cases, indicating that the model had not learned sharp preferences for where to attend (typically proper nouns, quantities, the central verb). It may also be the case that in an RNN, the best summary of each sentence is found at the end of the sentence (after all words have been considered), so there is little advantage to the variable-attention mechanism, particularly if in most cases it amounts to averaging over the latent representations at each word in the sentence.

In all the results reported in Table 7, the document length threshold was fixed as $L = 5$, meaning that documents were partitioned into subsequences of five sentences. It would be interesting to know whether the results depended on this parameter. Taking $L = 1$ would be identical to the standard RNNLM, run separately on each sentence. To test the effect of increasing L, an empirical comparison was performed between $L = 5$ and $L = 10$ with ccDCLM. Figure 22 shows the two curves on the PTB development set. The x-axis is the number of updates on ccDCLM with the

PTB training set. The y-axis is the mean per-token log-likelihood given by Equation 31 on the development set. As shown in this figure, $L = 10$ seems to learn more quickly per iteration in the beginning, although each iteration is more time-consuming, due to the need to backpropagate over longer documents. However, after a sufficient number of updates, the final performance results are nearly identical, with a slight advantage to the $L = 5$ setting. This suggests a tradeoff between the amount of contextual information and the ease of learning.

4.2.3.2 Local Coherence Evaluation

The long-term goal of coherence evaluation is to predict which texts are more coherent, and then to optimize for this criterion in multi-sentence generation tasks such as summarization and machine translation. A well-known proxy to this task is to try to automatically distinguish an original document from an alternative form in which the sentences are scrambled [5, 114]. Multi-sentence language models can be applied to this task directly, by determining whether the original document has a higher likelihood; no supervised training is necessary.

The specific experimental setup follows the same idea proposed by [5]. Considering limited number of documents in the PTB test set, I used bootstrapping [37] to give a robust model comparison. First, a new test set $\mathcal{D}^{(\ell)}$ is generated by sampling the documents from the original test set with replacement. Then, the sentences in each document $d \in \mathcal{D}^{(\ell)}$ of this new set was shuffled to get a pseudo-document d'. The combination of d and d' forms a single test example. Repeating the same procedure to produce 1,000 test sets, where each test set includes 155 document pairs. Since each test instance is a pairwise choice, a random baseline will have expected accuracy of 50%.

To evaluate the models proposed on this task, I used the configuration with the best development set perplexity, as shown in Table 11. The results of accuracy and

Table 8: Coherence evaluation on the PTB test set. The reported accuracies are calculated from 1,000 bootstrapping test sets (as explained in text).

	Accuracy	
Model	Mean (%)	Standard deviation (%)
Baselines		
1. RNNLM w/o sentence boundary (DRNNLM)	72.54	8.46
2. Hierarchical RNNLM (HRNNLM) [117]	75.32	4.42
Our models		
3. Attentional DCLM (ADCLM)†	75.51	4.12
4. Context-to-output DCLM (CODCLM)†*	81.72	3.81
5. Context-to-context DCLM (CCDCLM)†*	83.26	3.77

† significantly better than DRNNLM with p-value < 0.01
* significantly better than HRNNLM with p-value < 0.01

standard deviation are calculated over 1,000 resampled test sets. As shown in Table 8, the best accuracy is 83.26% given by CCDCLM, which also gives the smallest standard deviation 3.77%. Furthermore, all DCLM-based models significantly outperform the RNNLM with $p < 0.01$ given by a two-sample one-side z-test on the bootstrap samples. In addition, the CCDCLM and CODCLM are outperform the HRNNLM with $p < 0.01$ with statistic $z = 36.55$ and 31.26 respectively.

Unlike some prior work on coherence evaluation [114, 116, 117], this coherence evaluation approach does not require training on supervised data. Even though supervised training might therefore improve performance further, I would like to emphasize that the goal of this work is to make automatically-generated translations and summaries more coherent. Therefore, it is reasonable to avoid overfitting on this artificial proxy task.

4.3 Language Modeling with Discourse Relations

In last section, the DCLMs demonstrate the contribution of contextual information for word generation. The section will move one step further to consider the influence of discourse relations on generation. In general, discourse relations can be viewed as

the expectation of content in following sentences. The example discussed in subsection 1.3.2 illustrates that discourse relations further constrain the choice of words in generation.

This section provides a continuous work on word generation driven by discourse information. The new model utilizes a hybrid architecture that combines a RNNLM with a latent variable model over shallow discourse structure. The model learns a discriminatively-trained distributed representation of the local contextual features that drive word choice at the intra-sentence level, using techniques that are now state-of-the-art in language modeling [137]. However, the model treats shallow discourse structure — specifically, the relationships between pairs of adjacent sentences — as a latent variable. As a result, it can act as a discourse relation classifier besides a language model. Specifically:

- If trained to maximize the conditional likelihood of the discourse relations, it outperforms state-of-the-art methods for both implicit discourse relation classification in the Penn Discourse Treebank [169] and dialog act classification in Switchboard [93]. The model learns from both the discourse annotations as well as the language modeling objective, unlike previous recursive neural architectures that learn only from annotated discourse relations [82].
- If trained to maximize the joint likelihood of the discourse relations and the text, it can marginalize over discourse relations at test time, outperforming language models that do not account for discourse structure.

In contrast to recent work on continuous latent variables in recurrent neural networks [31], which require complex variational autoencoders to represent uncertainty over the latent variables, this model is simple to implement and train, requiring only minimal modifications to existing recurrent neural network architectures that are

implemented in commonly-used toolkits such as Theano, Torch, and CNN[1].

4.3.1 Shallow Discourse Relations

This work focuses on a class of *shallow discourse relations*, which hold between pairs of adjacent sentences (or utterances). These relations describe how the adjacent sentences are related: for example, they may be in CONTRAST, or the latter sentence may offer an answer to a question posed by the previous sentence. The assumption of this simple discourse structure is that *discourse relations only hold between adjacent text units*. An example is `unit1-discourse-unit2`, where `unit1` and `unit2` are two basic text units of this structure, and `discourse` represents the discourse information connecting these two text units. The discourse information can either be signaled by some lexical terms or inferred from semantics. Even it is simple, this structure still partially or even fully represents the typical discourse structure in some discourse theories on monologue and dialogue.

The first example of its connections is Lexicalized TAG theory for discourse (D-LTAG) proposed in [204] and her colleagues. D-LTAG employs a predicate-argument structure to model low-level discourse, where two arguments `Arg1` and `Arg2` are connected by a predicate explicitly or implicitly. A predicate in D-LTAG is a word or phrase called *discourse marker*. Discourse relations signaled by discourse markers explicitly are called explicit relations; otherwise, they are implicit relations. One argument is usually a clause or sentence. In minor cases, one argument can be several sentences or even a paragraph. Using the language of D-LTAG, the proposed simple discourse structure can be formulated as `Arg1-Marker-Arg2`. Based on the annotation scheme on explicit and implicit relations in the Penn Discourse Treebank (PDTB) [165], it covers all implicit cases and some explicit cases.

The similar connection to dialogue discourse theory is the Switchboard DAMSL

[1] `https://github.com/clab/cnn`

annotation scheme [92] on dialogue acts. A dialogue act (e.g., ACKNOWLEDGE) indicates the expectation of from one utterance `utter1` to the next `utter2` during a conversation. Based on the annotation scheme in the Switchboard dialogue act corpus, the simplified discourse structure basically covers all the cases with the realization `Utter1-Act-Utter2`.

4.3.2 Discourse Relation Language Models

This section presents the probabilistic neural model over sequences of words and shallow discourse relations. Discourse relations z_t are treated as latent variables, which are linked with a recurrent neural network over words in a *latent variable recurrent neural network* [31]. The model is formulated as a two-step generative story. In the first step, context information from the sentence $(t-1)$ is used to generate the discourse relation between sentences $(t-1)$ and t,

$$p(z_t \mid \boldsymbol{y}_{t-1}) = \text{softmax}\left(\boldsymbol{U}\boldsymbol{c}_{t-1} + \boldsymbol{b}\right), \tag{42}$$

where z_t is a random variable capturing the discourse relation between the two sentences, and $\boldsymbol{c}_{t-1}$ is a vector summary of the contextual information from sentence $(t-1)$, just as in the DCLM (section 4.2).

$$p(y_{t,n+1} \mid z_t,\, \boldsymbol{y}_{t,<n},\, \boldsymbol{y}_{t-1}) = g\Big(\underbrace{\mathbf{W}_o^{(z_t)}\boldsymbol{h}_{t,n}}_{\substack{\text{relation-specific} \\ \text{intra-sentential context}}} + \underbrace{\mathbf{W}_c^{(z_t)}\boldsymbol{c}_{t-1}}_{\substack{\text{relation-specific} \\ \text{inter-sentential context}}} + \underbrace{\boldsymbol{b}_o^{(z_t)}}_{\substack{\text{relation-specific} \\ \text{bias}}} \Big) \tag{43}$$

In the second step, the sentence $\boldsymbol{y}_t$ is generated, conditioning on the preceding sentence $\boldsymbol{y}_{t-1}$ and the discourse relation z_t:

$$p(\boldsymbol{y}_t \mid z_t, \boldsymbol{y}_{t-1}) = \prod_{n}^{N_t} p(y_{t,n} \mid \boldsymbol{y}_{t,<n}, \boldsymbol{y}_{t-1}, z_t), \tag{44}$$

The generative probability for the sentence $\boldsymbol{y}_t$ decomposes across tokens as usual (Equation 44). The per-token probabilities are shown in Equation 43. Discourse relations are incorporated by parameterizing the output matrices $\mathbf{W}_o^{(z_t)}$ and $\mathbf{W}_c^{(z_t)}$; depending on the discourse relation that holds between $(t-1)$ and t, these matrices will favor different parts of the embedding space. The bias term $\boldsymbol{b}_o^{(z_t)}$ is also parameterized by the discourse relation, so that each relation can favor specific words.

Overall, the joint probability of the text and discourse relations is,

$$p(\boldsymbol{y}_{1:T}, \boldsymbol{z}_{1:T}) = \prod_t^T \{p(z_t \mid \boldsymbol{y}_{t-1}) p(\boldsymbol{y}_t \mid z_t, \boldsymbol{y}_{t-1})\}. \tag{45}$$

If the discourse relations z_t are not observed, then this model is equivalent to a latent variable recurrent neural network (LVRNN). Connections to recent work on LVRNNs are discussed in section 4.1; the key difference is that the latent variables here correspond to linguistically meaningful elements, which are used to predict or marginalize, depending on the situation.

As proposed, the Discourse Relation Language Model has a large number of parameters. Let K, H and V be the input dimension, hidden dimension and the size of vocabulary in language modeling. The size of each prediction matrix $\mathbf{W}_o^{(z)}$ and $\mathbf{W}_c^{(z)}$ is $V \times H$; there are two such matrices for each possible discourse relation. The number of parameters is reduced by factoring each of these matrices into two components:

$$\mathbf{W}_o^{(z)} = \mathbf{W}_o \cdot \mathbf{V}^{(z)}, \quad \mathbf{W}_c^{(z)} = \mathbf{W}_c \cdot \mathbf{M}^{(z)}, \tag{46}$$

where $\mathbf{V}^{(z)}$ and $\mathbf{M}^{(z)}$ are relation-specific components for intra-sentential and inter-sentential contexts; the size of these matrices is $H \times H$, with $H \ll V$. The larger $V \times H$ matrices $\mathbf{W}_o$ and $\mathbf{W}_c$ are shared across all relations.

The discourse relation language model is carefully designed to decouple the discourse relations from each other, after conditioning on the words. It is clear that text documents and spoken dialogues have sequential discourse structures, and it

seems likely that modeling this structure could improve performance. In a traditional hidden Markov model (HMM) generative approach [186], modeling sequential dependencies is not difficult, because training reduces to relative frequency estimation. However, in the hybrid probabilistic-neural architecture proposed here, training is already expensive, due to the large number of parameters that must be estimated. Adding probabilistic couplings between adjacent discourse relations $\langle z_{t-1}, z_t \rangle$ would require the use of dynamic programming for both training and inference, increasing time complexity by a factor that is quadratic in the number of discourse relations.

Previous section also includes an alternative form of the document context language model called ccDCLM, in which the contextual information $\boldsymbol{c}_t$ impacts the hidden state $\boldsymbol{h}_{t+1}$, rather than going directly to the outputs $\boldsymbol{y}_{t+1}$. They obtain slightly better perplexity with this approach, which has fewer trainable parameters. However, this model would couple z_t with *all* subsequent sentences $\boldsymbol{y}_{>t}$, making prediction and marginalization of discourse relations considerably more challenging. Sequential Monte Carlo algorithms offer a possible solution [38], which may be considered in future work.

4.3.3 Inference

There are two possible inference scenarios: inference over discourse relations, conditioning on words; and inference over words, marginalizing over discourse relations.

Inference over Discourse Relations The probability of discourse relations given the sentences $p(\boldsymbol{z}_{1:T} \mid \boldsymbol{y}_{1:T})$ is decomposed into the product of probabilities of individual discourse relations conditioned on the adjacent sentences $\prod_t p(z_t \mid \boldsymbol{y}_t, \boldsymbol{y}_{t-1})$. These probabilities are computed by Bayes' rule:

$$p(z_t \mid \boldsymbol{y}_t, \boldsymbol{y}_{t-1}) = \frac{p(\boldsymbol{y}_t \mid z_t, \boldsymbol{y}_{t-1}) p(z_t \mid \boldsymbol{y}_{t-1})}{\sum_{z'} p(\boldsymbol{y}_t \mid z', \boldsymbol{y}_{t-1}) p(z' \mid \boldsymbol{y}_{t-1})}. \tag{47}$$

The terms in each product are given in Equations 42 and 44. Normalizing involves only a sum over a small finite number of discourse relations. Note that inference is easy in our case because all words are observed and there is no probabilistic coupling of the discourse relations.

Inference over Words In discourse-informed language modeling, inference needs to marginalize over discourse relations to compute the probability of a sequence of sentence $\boldsymbol{y}_{1:T}$, which can be written as,

$$p(\boldsymbol{y}_{1:T}) = \prod_{t}^{T} \sum_{z_t} p(z_t \mid \boldsymbol{y}_{t-1}) p(\boldsymbol{y}_t \mid z_t, \boldsymbol{y}_{t-1}), \tag{48}$$

because the word sequences are observed, decoupling each z_t from its neighbors z_{t+1} and z_{t-1}. This decoupling ensures that the overall marginal likelihood can be compuated as a product over local marginals.

4.3.4 Learning

The model can be trained in two ways: to maximize the joint probability $p(\boldsymbol{y}_{1:T}, \boldsymbol{z}_{1:T})$, or to maximize the conditional probability $p(\boldsymbol{z}_{1:T} \mid \boldsymbol{y}_{1:T})$. The joint training objective is more suitable for language modeling scenarios, and the conditional objective is better for discourse relation prediction.

Joint Likelihood Objective The joint likelihood objective function is directly adopted from the joint probability defined in Equation 45. The objective function for a single document with T sentences or utterances is,

$$\ell(\boldsymbol{\theta}) = \sum_{t}^{T} \log p(z_t \mid \boldsymbol{y}_{t-1}) + \sum_{n}^{N_t} \log p(y_{t,n} \mid \boldsymbol{y}_{t,<n}, \boldsymbol{y}_{t-1}, z_t), \tag{49}$$

where $\boldsymbol{\theta}$ represents the collection of all model parameters, including the parameters in the LSTM units and the word embeddings.

Maximizing the objective function $\ell(\boldsymbol{\theta})$ will jointly optimize the model on both language and discourse relation prediction. As such, it can be viewed as a form of

multi-task learning [25], where the model learns a shared representation that works well for discourse relation prediction and for language modeling. However, in practice, the large vocabulary size and number of tokens means that the language modeling part of the objective function tends to dominate. Therefore, an alternative objective is proposed if the model expects to focus more on discourse relation prediction.

Conditional Objective The training objective is specific to the discourse relation prediction task, and based on Equation 47 can be written as:

$$\ell_r(\boldsymbol{\theta}) = \sum_t^T \log p(z_t \mid \boldsymbol{y}_{t-1}) + \log p(\boldsymbol{y}_t \mid z_t, \boldsymbol{y}_{t-1}) - \log \sum_{z'} p(z' \mid \boldsymbol{y}_{t-1}) \times p(\boldsymbol{y}_t \mid z', \boldsymbol{y}_{t-1}) \tag{50}$$

The first line in Equation 50 is the same as $\ell(\boldsymbol{\theta})$, but the second line reflects the normalization over all possible values of z_t. This forces the objective function to attend specifically to the problem of maximizing the conditional likelihood of the discourse relations.

4.3.5 Evaluation

The model was evaluated on two benchmark datasets: (1) the Penn Discourse Treebank [165, PDTB], which is annotated on a corpus of Wall Street Journal articles; (2) the Switchboard dialogue act corpus [186, SWDA], which is annotated on a collections of phone conversations. Both corpora contain annotations of discourse relations and dialogue relations that hold between adjacent spans of text.

The Penn Discourse Treebank (PDTB) provides a low-level discourse annotation on written texts. As discussed in chapter 2, each discourse relation in the PDTB is annotated between two argument spans, `Arg1` and `Arg2`. There are two major types of relations: explicit and implicit. Explicit relations are signaled by discourse markers (e.g., "*however*", "*moreover*"), and the span of `Arg1` is almost totally unconstrained: it can range from a single clause to an entire paragraph, and need not

be adjacent to either `Arg2` nor the discourse marker. However, automatically classifying these relations is considered to be relatively easy, due to the constraints from the discourse marker itself [163]. In contrast, *implicit* discourse relations are annotated only between adjacent sentences, based on a semantic understanding of the discourse arguments. Automatically classifying these discourse relations is a challenging task [118, 162, 169, 82]. The evaluation task on the PDTB focuses on implicit discourse relations, leaving to the future work the question of how to apply the modeling framework to explicit discourse relations. During training, all relation types other than implicit (including explicit, EntRel, and NoRel) are collapsed into a single dummy relation type, which holds between all adjacent sentence pairs that do not share an implicit relation.

As in the prior work on first-level discourse relation identification (e.g., Park and Cardie, 2012), the evaluation used sections 2-20 of the PDTB as the training set, sections 0-1 as the development set for parameter tuning, and sections 21-22 for testing. Preprocessing includes lower-casing all tokens, and substituting all numbers with a special token "*NUM*". In addition, the 10,000 most frequent words from the training set were used to build the vocabulary, leaving low-frequency words replaced with a special token "*UNK*".

In prior work that focuses on detecting individual relations, balanced training sets are constructed so that there are an equal number of instances with and without each relation type [156, 13, 170]. This task targets the more challenging multi-way classification problem, so this strategy is not applicable; in any case, since the method deals with entire documents, it is not possible to balance the training set in this way.

The Switchboard Dialog Act Corpus (SWDA) is annotated on the Switchboard Corpus of human-human conversational telephone speech [56]. The annotations label each utterance with one of 42 possible speech acts, such as agree, hedge, and wh-question. Because these speech acts form the structure of the dialogue, most of them

pertain to both the preceding and succeeding utterances (e.g., AGREE). The SWDA corpus includes 1155 five-minute conversations. Standard split from [186] was adopted in the experiment, with 1,115 conversations for training and nineteen conversations for test. For parameter tuning, a randomly selected nineteen conversations from the training set was used as the development set. After parameter tuning, I retrained the model on the full training set with the selected configuration. I used the same preprocessing techniques here as in the PDTB.

A single-layer LSTM is used to build the recurrent architecture of the model, which is implemented in the CNN package[2]. Following prior work on RNN initialization [7], all parameters except the relation prediction parameters $\mathbf{U}$ and $\boldsymbol{b}$ are initialized with random values drawn from the range $[-\sqrt{6/(d_1+d_2)}, \sqrt{6/(d_1+d_2)}]$, where d_1 and d_2 are the input and output dimensions of the parameter matrix respectively. The matrix $\mathbf{U}$ is initialized with random numbers from $[-10^{-5}, 10^{-5}]$ and $\boldsymbol{b}$ is initialized to $\mathbf{0}$. Online learning was performed using AdaGrad [43] with initial learning rate $\lambda = 0.1$. Similarly to what I did on training DCLMs, I used norm clipping trick with a threshold of $\tau = 5.0$ [158] to avoid the exploding gradient problem. In addition, I also used value dropout [184] with rate 0.5, on the input $\mathbf{X}$, context vector $\boldsymbol{c}$ and hidden state $\boldsymbol{h}$, similar to the architecture proposed by [211]. The model includes two tunable hyper-parameters: the dimension of word representation K, the hidden dimension of LSTM unit H. In experiments, I considered the values $\{32, 48, 64, 96, 128\}$ for both K and H. For each corpus in experiments, the best combination of K and H is selected via grid search on the development set.

As the model can do discourse relation identification and language modeling. The evaluation here also is two-fold: (1) evaluating discourse relation prediction on both the PDTB and SWDA corpora, to show how multi-task learning in this model could help identifying discourse relations; (2) evaluating perplexity on the discourse

[2] `https://github.com/clab/cnn`

driven language modeling also on both corpora, to demonstrate whether incorporating discourse annotations at training time and then marginalizing them at test time can improve performance.

First, evaluating the model with implicit discourse relation prediction on the PDTB dataset. Most of the prior work on first-level discourse relation prediction focuses on the "one-versus-all" binary classification setting, but I would like to attack the more general four-way classification problem, as performed by [169]. Specifically, the model is compared against the following methods:

Most common class The most common first level PDTB relation is EXPANSION.

Rutherford and Xue (2015) [169] build a set of feature-rich classifiers on the PDTB, and then augment these classifiers with additional automatically-labeled training instances. I compare against their published results, which are state-of-the-art.

Ji and Eisenstein (2015) [82] This is the model I discussed in chapter 3. But the experimental setting is different here, so I rerun the system using same setting described above.

As shown in Table 9, the conditionally-trained discourse relation language model (DRLM) outperforms all alternatives, on both metrics. While the jointly-trained model is at the same level as the previous state-of-the-art, conditional training provides a significant additional advantage.

For the second discourse relevant evaluation, I use dialog act tagging on the conversation data. Dialogue act tagging has been widely studied in both NLP and speech communities. Here, I follow the setup used in [186] to conduct experiments, and adopt the following systems for comparison:

Most common class The most common dialog act is STATEMENT-NON-OPINION.

Table 9: Multiclass relation identification on the first-level PDTB relations.

Model	Accuracy	Macro F_1
Baseline		
1. Most common class	54.7	—
Prior work		
2. Rutherford and Xue (2015) [169]	55.0	38.4
3. Rutherford and Xue (2015) [169] with extra training data	57.1	40.5
4. Ji and Eisenstein (2015) [82]	56.4	40.0
Our work - DRLM		
5. Joint training	57.1	40.5
6. Conditional training	59.5*	42.3

* significantly better than lines 2 and 4 with $p < 0.05$

Stolcke *et al.***(2000) [186]** employ a hidden Markov model, with each HMM state corresponding to a dialogue act.

Kalchbrenner and Blunsom (2013) [93] employ a complex neural architecture, with a convolutional network at each utterance and a recurrent network over the length of the dialog. To my best knowledge, this model attains state-of-the-art accuracy on this task, outperforming other prior work such as [203, 141].

As shown in Table 10, the conditionally-trained discourse relation language model (DRLM) outperforms all competitive systems on this task. All comparisons are against published results, and Macro-F_1 scores are not available. Accuracy is more reliable on this evaluation, since no single class dominates, unlike the PDTB task.

As discussed before, since the joint model DRLM is on both discourse and language modeling, it can also function as a language model, assigning probabilities to sequences of words while marginalizing over discourse relations. To determine whether discourse-aware language modeling can improve performance, I compare it against the following systems:

RNNLM+LSTM This is the same basic architecture as the RNNLM proposed by Mikolov *et al.*[137], which was shown to outperform a Kneser-Ney smoothed

Table 10: The results of dialogue act tagging.

1. Model	Accuracy
Baseline	
2. Most common class	31.5
Prior work	
3. Stolcke *et al.* (2000) [186]	71.0
4. Kalchbrenner and Blunsom (2013) [93]	73.9
Our work - DRLM	
5. Joint training	74.0
6. Conditional training	**77.0***

* significantly better than line 4 with $p < 0.01$

5-gram model on modeling Wall Street Journal text. Following the model in [211], I replaced the Sigmoid nonlinearity with a LSTM.

DCLM The model is also compared against the DCLMs discussed in section 4.2. More specific, I used the CODCLM, which is identical to the current modeling approach, except that it is not parameterized by discourse relations. This model achieves strong results on language modeling for small and medium-sized corpora, outperforming RNNLM+LSTM.

The perplexities of language modeling on the PDTB and the SWDA are summarized in Table 11. The comparison between line 1 and line 2 shows the benefit of considering multi-sentence context information on language modeling. Line 3 shows that adding discourse relation information yields further improvements for both datasets. Recall that discourse relations in the test documents are marginalized out, so no annotations are required for the test set; the improvements are due to the disambiguating power of discourse relations in the training set.

Because training on DRLM and LVRNN-SOFT requires discourse annotations, this approach does not scale to the large datasets typically used in language modeling. As a consequence, the results obtained here are somewhat academic, from the perspective of practical language modeling. Nonetheless, the positive results here motivate

Table 11: Language model perplexities (PPLX), lower is better. The model dimensions K and H that gave best performance on the dev set are also shown.

	PDTB			SWDA		
Model	K	H	PPLX	K	H	PPLX
Baseline						
1. RNNLM	96	128	117.8	128	96	56.0
2. DCLM	96	96	112.2	96	96	45.3
Our work						
3. DrLM	64	96	108.3	128	64	39.6

the investigation of training procedures that are also capable of marginalizing over discourse relations.

4.4 Discussion

As shown in this chapter, contextual information beyond the sentence boundary is essential to document-level text generation and coherence evaluation. The set of document-context language models (DCLMs; section 4.2) proposed in this chapter provides various approaches to incorporate contextual information from preceding texts. Empirical evaluation with perplexity shows that the DCLMs give better word prediction as language models in comparison with conventional RNNLMs and also good performance on unsupervised coherence assessment.

In addition, a probabilistic neural model (DrLM) presented in section 4.3 to unite sequences of words and shallow discourse relations between adjacent sequences into a framework. This model combines positive aspects of neural network architectures with probabilistic graphical models: it can learn discriminatively-trained vector representations, while maintaining a probabilistic representation of shallow discourse relations. This method can be applied as a language model, marginalizing over discourse relations on the test data. It also outperforms state-of-the-art systems in two discourse relation detection tasks.

CHAPTER V

SEMANTIC REPRESENTATION LEARNING WITH DISTANT SUPERVISION

Work described in this chapter was undertaken in collaboration with Jacob Eisenstein and Gongbo Zhang, published at EMNLP 2015 [84]

In chapter 3 and chapter 4, I discuss the possibility of utilizing supervision information for representation learning and discourse processing. The idea of representation learning has significantly improved the performance on several discourse related tasks, such as RST-style discourse parsing, implicit discourse relation identification and discourse driven language modeling. Obviously, the prerequisite of using these models in practice is the existence of data with discourse annotation. Unfortunately, annotating discourse information requires some necessary pre-training with annotators, and therefore may not be possible to employ the crowd-sourcing method. For example, the largest well-known discourse corpus is the Penn Discourse Treebank [165]. It took nearly 4 years to annotate the first version, and then 2 more years to get 2,159 documents annotated in the second version [165]. To eliminate the dependency of discourse annotation, it is necessary to study the possibility of unsupervised discourse processing or processing with distant supervision.

Unfortunately, prior work on discourse processing with distant supervision constantly presents some negative results. For example, Sporleder and Lascarides (2008) [183] show that models trained on explicitly marked examples generalize poorly to implicit relation identification. They argued that explicit and implicit examples may be linguistically dissimilar, as writers tend to avoid discourse connectives if the discourse relation could be inferred from context [58]. More prior work will be discussed in

section 5.1.

In this chapter, I try to provide a solution to this problem from the perspective of domain adaptation. Specifically, I argue that the reason that automatically-labeled examples generalize poorly is due to domain mismatch from the explicit relations (**source** domain) to the implicit relations (**target** domain). To close the gap with in-domain relation identification, two simple domain adaptation methods are used here: (1) feature representation learning: mapping the source domain and target domain to a shared latent feature space; (2) resampling: modifying the relation distribution in the explicit relations to match the distribution over implicit relations.

5.1 Prior Work

There is a limited prior work on discourse processing with distant supervision. Marcu and Echihabi [132] train a classifier for implicit intra-sentence discourse relations from explicitly-marked examples in the RST-DT, where the relations are automatically labeled by their discourse connectives: for example, labeling the relation as CONTRAST if the connective is *but*. However, Sporleder and Lascarides [183] later argue that explicitly marked relations are too different from implicit relations to serve as an adequate supervision signal, obtaining negative results in segmented discourse representation theory (SDRT) relations.

Recently, more relevant work has focused on the Penn Discourse Treebank (PDTB), using explicitly-marked relations to supplement, rather than replace, a labeled corpus of implicit relations. For example, Biran and McKeown [13] collect word pairs from arguments of explicit examples to help the supervised learning on implicit relation identification. Lan *et al.*[105] present a multi-task learning framework, using explicit relation identification as auxiliary tasks to help main task on implicit relation identification. Rutherford and Xue [169] explore several selection heuristics for adding automatically-labeled examples from Gigaword to their system for implicit relation

detection, obtaining a 2% improvement in Macro-F_1. This work differs from these previous efforts in that we focus exclusively on training from automatically-labeled explicit instances, rather than supplementing a training set of manually-labeled implicit examples.

Learning good feature representations [6] and reducing mismatched label distributions [88] are two main ways to make a domain adaptation task successful. Structural correspondence learning is an early example of representation learning for domain adaptation [18]. Instead, this work is built on the more computationally tractable approach of marginalized denoising autoencoders [27]. Another simple domain adaptation technique is instance weighting [85], which is used for correcting label distribution mismatch. This work employs a simpler approach of resampling the source domain according to an estimate of the target domain label distribution.

5.2 Domain Adaptation for Implicit Relation Identification

Two domain adaptation techniques are discussed in this section specifically for the relation identification problem.

5.2.1 Learning feature representation

The goal of feature representation learning is to obtain dense features that capture feature correlations between the source and target domains. Denoising autoencoders [55] do this by first "corrupting" the original data, $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$ into $\tilde{\boldsymbol{x}}_1, \ldots, \tilde{\boldsymbol{x}}_n$, either by adding Gaussian noise (in the case of real-valued data) or by randomly zeroing out features (in the case of binary data). Then, a function is learned to reconstruct the original data, thereby capturing feature correlations and improving resilience to domain shift.

Chen *et al.*[27] propose a particularly simple and elegant form of denoising autencoder, by marginalizing over the noising process. Their single-layer marginalized

denoising autoencoder (mDA) solves the following problem:

$$\min_{\mathbf{W}} E_{\tilde{\boldsymbol{x}}_i|\boldsymbol{x}_i}[\|\boldsymbol{x}_i - \mathbf{W}\tilde{\boldsymbol{x}}_i\|^2] \tag{51}$$

where the parameter $\mathbf{W} \in \mathbb{R}^{d\times d}$ is a projection matrix. After learning the projection matrix, $\tanh(\mathbf{W}\boldsymbol{x})$ can be used as the representation for our relation identification task.

Usually, $\boldsymbol{x}_i \in \mathbb{R}^d$ is a sparse vector with more than 10^5 dimensions. Solving the optimization problem defined in equation 51 will produce a $d \times d$ dense matrix $\mathbf{W}$, and is prohibitively expensive. This work employs the trick proposed by [18] to select κ *pivot* features to be reconstructed, and split all features into non-overlapping subsets of size $\leq K$. Then, a set of projection matrices are learned, so as to transform each feature subset to the pivot feature set. The final projection matrix $\mathbf{W}$ is the stack of all projection matrices learned from the feature subsets.

5.2.2 Resampling with minimal supervision

There is a notable mismatch between the relation distributions for implicit and explicitly-marked discourse relations in the Penn Discourse Treebank: as shown in Figure 23, the EXPANSION and CONTINGENCY relations comprise a greater share of the implicit relations, while the TEMPORAL and COMPARISON relations comprise a greater share of the explicitly-marked discourse relations. Such label distribution mismatches can be a major source of transfer loss across domains, and therefore, reducing this mismatch can be an easy way to obtain performance gains in domain adaptation [88]. Specifically, the goal here is to modify the relation distribution in the source domain (explicitly-marked relations) and make it as similar as possible to the target domain (implicit relations). Given the label distribution from the target domain, the training examples are resampled from the source domain with replacement, in order to match the label distribution in the target domain. As this requires the label distribution from the target domain, it is no longer purely unsupervised

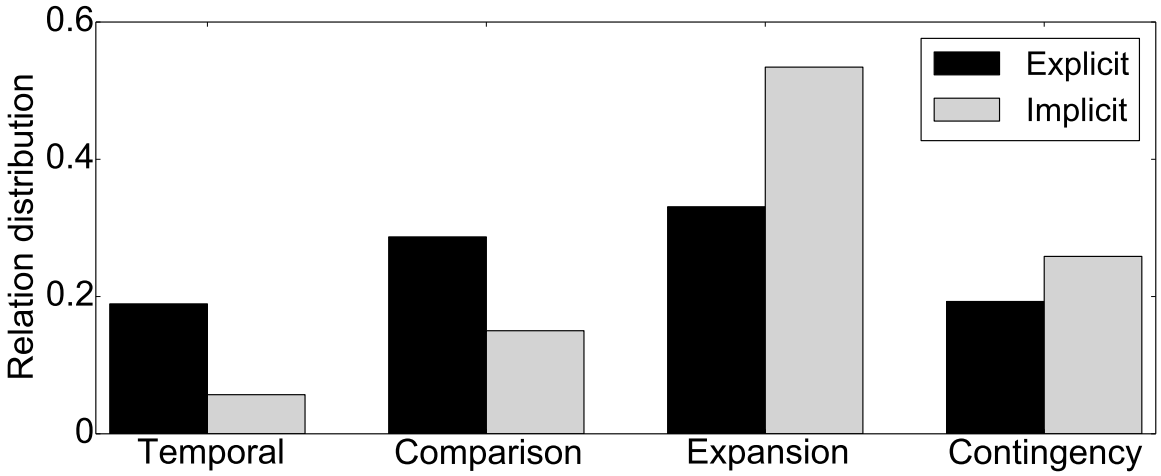


Figure 23: The relation distributions of training examples from the source domain (explicitly-marked relations) and target domain (implicit relations) in the PDTB.

domain adaptation; instead, it is named as *resampling with distant supervision*.

It may also be desirable to ensure that the source and target training instances are similar in terms of their observed features; this is the idea behind the *instance weighting* approach to domain adaptation [85]. This idea suggests that sampled instances from the source domain have a cosine similarity of at least τ with at least one target domain instance [169].

5.2.3 Evaluation

The experiments test the utility of the two domain adaptation methods, using the Penn Discourse Treebank [165] and some extra-training data collected from a external resource.

The test examples are implicit relation instances from section 21-22 in the PDTB. For the domain adaptation setting, the training set consists of the explicitly-marked examples extracted from sections 02-20 and 23-24, and the development set consists of the explicit relations from sections 21-22. All relations in the explicit examples are automatically labeled by using the connective-to-relation mapping from Table 2 in [167], where only keeping the majority relation type for every connective. For each identified connective, the evaluation uses its annotated arguments in the PDTB. To

give an upper bound, the evaluation employs an in-domain discourse relation classifier, using the implicit examples in sections 02-20 and 23-24 as the training set, and using sections 00-01 as the development set. Following prior work [162, 156, 13], relation identification on the PDTB considered the first-level discourse relations in the PDTB — Temporal (TEMP.), Comparison (COMP.), Expansion (EXP.) and Contingency (CONT.). In total, four binary classifiers were trained and report F_1 score on each binary classification task.

The true power of learning from automatically labeled examples is that we could leverage much larger datasets than hand-annotated corpora such as the Penn Discourse Treebank. To test this idea, my collaborator collected 1,000 news articles from CNN.com as extra training data. Explicitly-marked discourse relations from this data are automatically extracted by matching the PDTB discourse connectives [167]. This data also needs to extract the arguments of the identified connectives: for every identified connective, the sentence following this connective is labeled as `Arg2` and the preceding sentence is labeled as `Arg1`, as suggested in [13]. A pilot study in this work shows larger amounts of additional training data yielded no further improvements, which is consistent with the recent results in [169].

For relation identification, a linear support vector machine with a penalty parameter C [47] is used as the classification model. In addition, the model includes four tunable parameters: the number of pivot features κ, the size of the feature subset K, the noise level for the denoising autoencoder q, and the cosine similarity threshold for resampling τ. I considered $\kappa \in \{1000, 2000, 3000\}$ for pivot features and $C \in \{0.001, 0.01, 0.1, 1.0, 10.0\}$ for penalty parameters, $q \in \{0.90, 0.95, 0.99\}$ for noise levels. To reduce the free parameters, I set $K = 5\kappa$ and simply fix the cosine similarity threshold $\tau = 0.85$; Preliminary studies found that results are not sensitive to the value of τ across a range of values.

All features are motivated by prior work on implicit discourse relation classification: from each training example with two arguments, I extracted the following features for representation learning

Lexical features including word pairs, the first and last words from both arguments [162];

Syntactic features including production rules from each argument, and the shared production rules between two arguments [118];

Other features including modality, Inquirer tags, Levin verb classes, and argument polarity [156].

I used the Stanford CoreNLP Toolkit to obtain syntactic annotations [127] and reimplemented the feature extractor as closely as possible to the cited works.

The FULL feature set for domain adaptation is constructed by collecting all features from the training set, and then removing features that occur fewer than ten times. The PIVOT feature set includes κ high-frequency features from the FULL feature set. To focus on testing the domain adaptation techniques, I used the same FULL and PIVOT set for all four relations, and leave feature set optimization for each relation as a future work [156]. To obtain the upper bound, the same feature categories and frequency threshold were used to extract features from the in-domain data, hand-annotated implicit discourse relations. The representations from mDA were concatenated with the original surface feature representations of the same examples as the final feature vectors for relation identification.

The experiment starts with surface feature representations as baselines, then incorporates the two domain adaptation techniques incrementally. As shown in line 2 of Table 13, the performance is poor if directly applying a model trained on the explicit examples with the FULL feature set, which is consistent with the observations of [183]: there is a 10.28% absolute reduction on average F_1 score from the upper bound

Table 12: Performance of cross-domain learning for implicit discourse relation identification.

			Relations				
Surface Features	+Rep. Learning	+Resampling	TEMP.	COMP.	EXP.	CONT.	Average F_1
Implicit → Implicit							
1. FULL			24.15	28.87	68.84	43.45	41.32
Explicit [PDTB] → Implicit							
2. FULL	No	No	17.13	20.54	50.55	36.14	31.04
3. FULL	No	Yes	15.38	23.88	62.04	35.29	34.14
4. FULL	Yes	No	17.53	22.77	50.85	36.43	31.90
5. FULL	Yes	Yes	17.05	22.00	63.51	38.23	35.20
6. PIVOT	No	No	17.33	23.89	53.53	36.22	32.74
7. PIVOT	No	Yes	17.73	25.39	62.65	36.02	35.44
8. PIVOT	Yes	No	18.66	25.86	63.37	38.87	36.69
9. PIVOT	Yes	Yes	19.26	25.74	68.08	41.39	38.62
Explicit [PDTB + CNN] → Implicit							
10. PIVOT	Yes	Yes	20.35	26.32	68.92	42.25	39.46

obtained with in-domain supervision (line 1). With mDA, the overall performance increases by 0.86% (line 4); resampling gives a further 4.16% improvement mainly because of the performance gain on the EXP. relation (line 5). The resampling method itself (line 3) also gives a better overall performance then mDA (line 4). However, the F_1 scores on the TEMP. and CONT. are even worse than the baseline (line 2).

Surface representations with the FULL feature set were found to cause serious overfitting in the experiments. A possible way to deal with this problem is to only use κ pivot features. As shown in line 6, it gives a stronger baseline of the cross-domain relation identification, as shown in line 6. Then, by incorporating resampling and feature representation learning individually, the average F_1 increases from 32.74% to 35.44% (line 7) and 36.69% (line 8) respectively. The combination of these two domain adaptation techniques boosts the average F_1 further to 38.62% (line 9). The additional CNN training data further improves performance to 39.46% (line 10). This represents an 8.42% improvement of average F_1 from the original result (line 2), for more than 80% reduction on the transfer loss incurred by training on explicit discourse relations.

An additional experiment is to use automatic argument extraction in both the PDTB and the CNN data, which would correspond to more truly unsupervised domain adaptation. (Recall that in the CNN data, we used adjacent sentences as argument spans, while in the PDTB data, we use expert annotations.) When using adjacent sentences as argument spans in both datasets, the average F_1 is 38.52% for the combination of representation learning and resampling. Compared to line 10, this is a 0.94% performance drop, indicating the importance of argument identification in the PDTB data.

5.3 Discussion

This chapter present a preliminary study on discourse processing with distant supervision, mainly focuses on how explicit relation pairs can be used to help predict relations holding in implicit pairs. The core idea discussed in this chapter is unsupervised feature representation learning with marginalized denoising autoencoder, and a resampling method built on the top of representation learning. Experiments on the PDTB show the combination of these two methods eliminates more than 80% of the transfer loss caused by training on explicit examples, increasing average F_1 from 31% to 39.5%, against a supervised upper bound of 41.3%. Honestly, the proposed method in this chapter did not ultimately solve the problem of unsupervised discourse processing — there is still performance gap between "in-domain" and "cross-domain" implicit relation identification. However, the idea representation learning (together with resampling) does reduces the performance gap notably. Compare to the prior work, I still consider this work is a success on this research direction.

CHAPTER VI

APPLICATIONS OF DISCOURSE INFORMATION

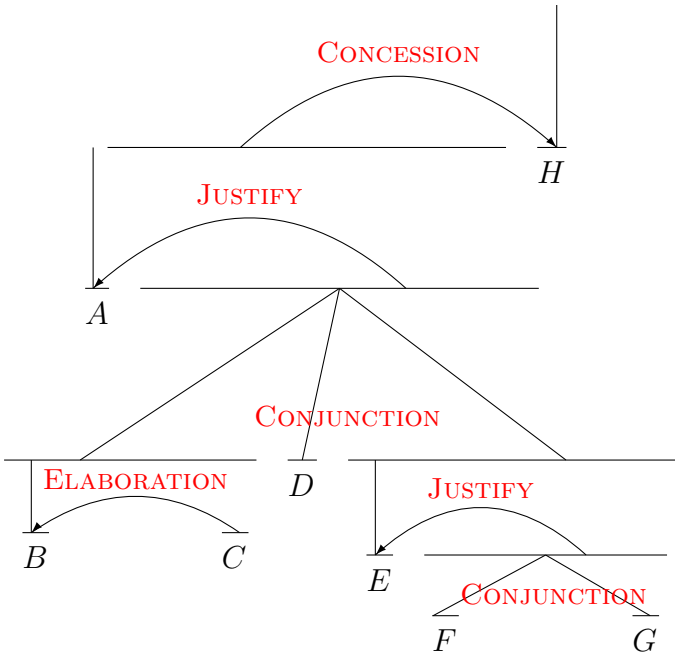
Work described in this chapter was undertaken in collaboration with Parminder Bhatia, Kevin Duh, Chris Dyer and Jacob Eisenstein. The work on sentiment analysis was published at EMNLP 2015 [12].

In this chapter, I discuss two discourse-relevant applications, to illustrate the merit of discourse information in natural language processing. The work on document-level sentiment analysis in section 6.1 shows the value of discourse structures on identifying the major opinion from a coherent text. The machine translation task in section 6.2 demonstrates the value of contextual information in document-level machine translation.

6.1 Discourse Information for Sentiment Analysis

Sentiment analysis and opinion mining are among the most widely-used applications of language technology, impacting both industry and a variety of other academic disciplines [48, 120, 154]. Research on sentiment analysis is still dominated by bag-of-words approaches, and attempts to include additional linguistic context typically stop at the sentence level [179]. Since document-level opinion mining inherently involves multi-sentence texts, it is a place where discourse structure definitely has a role to play.

A classic example of the potential relevance of discourse to sentiment analysis is shown in Figure 24. In this review on the film *The Last Samurai*, the positive sentiment words are more than the negative sentiment words. But the discourse structure — indicated with RST — clearly favors the final sentence, whose polarity is negative. This example is illustrative in more than one way: it was originally



[It could have been a great movie]^{A} [It does have beautiful scenery,]^{B} [some of the best since Lord of the Rings.]^{C} [The acting is well done,]^{D} [and I really liked the son of the leader of the Samurai.]^{E} [He was a likable chap,]^{F} [and I hated to see him die.]^{G} [But, other than all that, this movie is nothing more than hidden rip-offs.]^{H}

Figure 24: A Example of the RST structure for document-level sentiment analysis (adapted from (Voll and Taboada, 2007) [198]).

identified by Voll and Taboada [198], who found that *manually-annotated* RST parse trees improved lexicon-based sentiment analysis, but that automatically-generated parses from the SPADE parser [182], which was then state-of-the-art, did not.

Based on the progress of RST parsing presented in chapter 3, it is reasonable to revisit this topic. Since this time, RST discourse parsing has improved considerably, with the best systems now yielding $5-10\%$ greater raw accuracy than SPADE, depending on the metric. Therefore it is the time to reconsider the effectiveness of RST for document-level sentiment analysis. This section presents two ways of combining RST discourse parses with sentiment analysis.

- Reweighting the contribution of each discourse unit, based on its position in a dependency-like representation of the discourse structure. Such weights can be

defined using a simple function, or learned from a small of data.

- Recursively propagating sentiment up through the RST parse, in an architecture inspired by recursive neural networks [173, 177].

The methods are both simple and can be used in combination with an off-the-shelf discourse parser, for example, combining with either a lexicon-based sentiment analyzer or a trained classifier.

6.1.1 Prior Work on Discourse Information for Sentiment Analysis

The efforts to incorporate RST into sentiment analysis have often focused on intra-sentential discourse relations [65, 212, 28], rather than relations over the entire document. Wang *et al.* [201] address sentiment analysis in Chinese. Lacking a discourse parser, they focus on explicit connectives, using a strategy that is related to the discourse depth reweighting. Wang and Wu [200] use manually-annotated discourse parses in combination with a sentiment lexicon, which is automatically updated based on the discourse structure. Zirn *et al.* [214] use an RST parser in a Markov Logic Network, with the goal of making polarity predictions at the sub-sentence level, rather than improving document-level prediction. None of the prior work considers the sort of recurrent compositional model presented here.

An alternative to RST is to incorporate shallow discourse structures, such as the relations from the Penn Discourse Treebank (PDTB). PDTB relations were shown to improve sentence-level sentiment analysis [180], and were incorporated in a model of sentiment flow [199]. PDTB relations are often signaled with explicit discourse connectives, and these may be used as a feature [192, 108] or as posterior constraints [207]. This prior work on discourse relations within sentences and between adjacent sentences can be viewed as complementary to the focus on higher-level discourse relations across the entire document.

There are unfortunately few possibilities for direct comparison of the approach

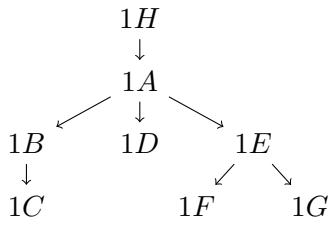


Figure 25: Dependency-based discourse tree representation of the discourse in Figure 24

discussed in this section against prior work. Heerschop *et al.* [65] and Wachsmuth *et al.* [199] also employ the Pang and Lee dataset [153], but neither of their results are directly comparable: [65] exclude documents that SPADE fails to parse, and [199] evaluates only on individual sentences rather than entire documents. The only possible direct comparison is with very recent work from Hogenboom *et al.* [74], who employ a weighting scheme that is similar to the approach described in Section 6.1.2. They evaluate on the Pang and Lee data, and consider only lexicon-based sentiment analysis, obtaining document-level accuracies between 65% (for the baseline) and 72% (for their best discourse-augmented system).

6.1.2 Discourse depth reweighting

The first approach to incorporating discourse information into sentiment analysis is based on quantifying the importance of each EDU in terms of its position in the discourse structure. Take the example illustrated in Figure 24, EDU H is considered as the most important text unit of this review. For the rest EDUs, their importance depends on their distance with H. To compute the distance efficiently, the *dependency-based discourse tree* (DEP-DT) formulation from prior work [69] is employed here. The DEP-DT formalism converts the constituent-like RST tree into a directed graph over elementary discourse units (EDUs), in a process that is a close analogue of the transformation of a headed syntactic constituent parse to a syntactic dependency graph [103].

The DEP-DT representation of the discourse in Figure 24 is shown in Figure 25. The graph is constructed by propagating information up the RST tree; if an EDU e_i is the satellite in a discourse relation headed by e_j, then there is a dependency edge from e_j to e_i. Thus, the depth of each EDU is the number of times in which it is embedded in the satellite of a discourse relation. The exact algorithm for constructing DEP-DTs is given in [69].

Given this representation, a simple linear function is constructed for weighting the contribution of the EDU at depth d_i:

$$\lambda_i = \max(0.5, 1 - d_i/6). \tag{52}$$

Thus, at $d_i = 0$, $\lambda_i = 1$, and at $d_i \geq 3$, $\lambda_i = 0.5$. This linear truncated weighting function is manually designed. The intuition behind this function is that the importance of an EDU linearly decreases with its depth.

Assume each EDU contributes a prediction $\psi_i = \boldsymbol{\theta}^\top \boldsymbol{w}_i$, where $\boldsymbol{w}_i$ is the bag-of-words vector, and θ is a vector of weights, which may be either learned or specified by a sentiment lexicon. Then the overall prediction for a document is given by,

$$\Psi = \sum_i \lambda_i (\boldsymbol{\theta}^\top \boldsymbol{w}_i) = \boldsymbol{\theta}^\top (\sum_i \lambda_i \boldsymbol{w}_i). \tag{53}$$

Evaluation For evaluation, this approach was combined with both lexicon-based and classification-based sentiment analysis. Specifically, it used the lexicon of [206], and set $\theta_j = 1$ for words marked positive, and $\theta_j = -1$ for words marked negative. For classification-based analysis, $\boldsymbol{\theta}$ is equal to the weights obtained by training a logistic regression classifier, tuning the regularization coefficient on held-out data.

Results of this method are shown in Table 13. As seen in the comparison between lines B1 and D1, discourse depth weighting offers substantial improvements over the bag-of-words approach for lexicon-based sentiment analysis, with raw improvements of $4 - 5\%$. Given the simplicity of this approach — which requires only a sentiment

Table 13: Sentiment classification accuracies on two movie review datasets [153, 179].

	Pang & Lee (2004)	Socher *et al.* (2013)
Baselines		
B1. Lexicon	68.3	74.9
B2. Classifier	82.4	81.5
Discourse depth weighting		
D1. Lexicon	72.6	78.9
D2. Classifier	82.9	82.0
Rhetorical recursive neural network		
R1. No relations	83.4	85.5
R2. With relations	84.1	85.6

lexicon and a discourse parser — it is recommended for lexicon-based sentiment analysis at the document level. However, the improvements for the classification-based models are considerably smaller, less than 1% in both datasets.

6.1.3 Rhetorical Recursive Neural Networks

Discourse-depth reweighting offers significant improvements for lexicon-based sentiment analysis, but the improvements over the more accurate classification-based method are meager. Therefore it is reasonable have a data-driven approach for combining sentiment analysis with rhetorical structure theory, based on recursive neural networks [177]. The idea is simple: recursively propagate sentiment scores up the RST tree, until the root of the document is reached. For nucleus-satellite discourse relations, it defines

$$\Psi_i = \tanh(K_n^{(r_i)}\Psi_{n(i)} + K_s^{(r_i)}\Psi_{s(i)}), \tag{54}$$

where i indexes a discourse unit composed from relation r_i, $n(i)$ indicates its nucleus, and $s(i)$ indicates its satellite. Returning to the example in Figure 24, the sentiment score for the discourse unit obtained by combining $1B$ and $1C$ is obtained from $\tanh(K_n^{(\text{elaboration})}\Psi_{1B} + K_s^{(\text{elaboration})}\Psi_{1C})$. Similarly, for multinuclear relations, it has

$$\Psi_i = \tanh(\sum_{j \in n(i)} K_n^{(r_i)}\Psi_j). \tag{55}$$

In the base case, each elementary discourse unit's sentiment is constructed from its bag-of-words, $\Psi_i = \boldsymbol{\theta}^\top \boldsymbol{w}_i$. Because the structure of each document is different, the network architecture varies in each example; nonetheless, the parameters can be reused across all instances.

This approach is called a Rhetorical Recursive Neural Network (R2N2). It is reminiscent of the compositional model proposed by Socher *et al.* [179], where composition is over the constituents of the syntactic parse of a sentence rather than the units of a discourse. A crucial difference in R2N2s is that the elements Ψ and K are *scalars*: it does not attempt to learn a latent distributed representation of the sub-document units. This is because discourse units typically comprise multiple words, so that accurate analysis of the sentiment for EDUs is not so difficult as accurate analysis of individual words. The scores for individual discourse units can be computed from a bag-of-words classifier, or, from a more complex model such as a recursive or recurrent neural network (as discussed in chapter 3).

While this neural network structure captures the idea of compositionality over the RST tree, the most deeply embedded discourse units can be heavily down-weighted by the recursive composition (assuming $K_s < K_n$). In the most extreme case of a right-branching or left-branching structure, the recursive operator may be applied N times to the most deeply embedded EDU. On the contrary, discourse depth reweighting applies a uniform weight of 0.5 to all discourse units with depth ≥ 3. This approach adds an additional component to the network architecture, capturing the bag-of-words for the entire document. Thus, at the root node it has

$$\Psi_{\text{doc}} = \gamma \boldsymbol{\theta}^\top (\sum_i \boldsymbol{w}_i) + \Psi_{\text{rst-root}}, \tag{56}$$

with $\Psi_{\text{rst-root}}$ defined recursively from Equation 54 and Equation 55, $\boldsymbol{\theta}$ indicating the vector of per-word weights, and the scalar γ controlling the tradeoff between these two components.

Learning R2N2 is trained by backpropagating from a hinge loss objective. Assuming $y_t \in \{-1, 1\}$ for each document t, the loss is defined as $L_t = (1 - y_t \Psi_{\text{doc},t})_+$. From this loss, the gradients on the parameters are obtained via backpropagation through structure [57]. Training is performed using stochastic gradient descent. For simplicity, this focuses on the distinction between contrastive and non-contrastive relations [214]. The set of contrastive relations includes CONTRAST, COMPARISON, ANTITHESIS, ANTITHESIS-E, CONSEQUENCE-S, CONCESSION, and PROBLEM-SOLUTION.

Evaluation Results for this approach are shown in lines R1 and R2 of Table 13. Even without distinguishing between discourse relations, an improvement is already obtained of more than 3% accuracy on the Stanford data, and 0.5% on the smaller Pang and Lee dataset. Adding sensitivity to discourse relations (distinguishing $K^{(r)}$ for contrastive and non-contrastive relations) offers further improvements on the Pang & Lee data, outperforming the baseline classifier (D2) by 1.3%. The accuracy of discourse relation detection is only 60% for even the best systems [81], which may help to explain why relations do not offer a more substantial boost. Given the limited amount of annotated data on both discourse structure and sentiment polarity, it is almost impossible to evaluate on gold RST parses. For example, the SFU Review Corpus [188] of 30 review texts offers a starting point, which is probably too small to train a competitive sentiment analysis system.

6.1.4 Conclusion

Sentiment polarity analysis has typically relied on a "preponderance of evidence" strategy, hoping that the words or sentences representing the overall polarity will outweigh those representing counterpoints or rhetorical concessions. However, with the availability of off-the-shelf RST discourse parsers, it is now easy to include document-level structure in sentiment analysis. This work shows that a simple reweighting approach offers robust advantages in lexicon-based sentiment analysis, and a recursive

neural network can substantially outperform a bag-of-words classifier.

6.2 Discourse Information of Document-level MT

Section 4.2 shows that a language model is capable of incorporating contextual information from preceding text. DCLMs proposed in section 4.2 are tested with a coherence evaluation task [5] and show the superior performance on that task. In this section, I would like to investigate a similar coherence-relevant problem in the machine translation scenario. Specifically, how to combine DCLM features to obtain better document-level translations? Note that, even at the sentence level, translation decoding is NP-complete [96]. Translating at the document level is thus certain to require approximation. To simplify the complexity of direct translation on the document level, I assume that each sentence in the source language already has a collection of translated hypotheses in the target language. Then, the document-level translation problem studied in this section is reduced into a sentence-level decoding problem: for each sentence, select the best hypothesis such that the whole translated document has a better coherence.

This section provides a preliminary study on this decoding problem. For the convenience of implementation, a greedy decoder is used to pick the best hypothesis based on DCLM features (together with some other features). A better decoding (e.g.; Viterbi decoding [15]) will be left as future work.

6.2.1 Prior work

The problem of discourse information for machine translation has been investigated for more than 20 years. For example, a paper by Mitkov [145] in 1993 considers the problem of how discourse relations can be used for document-level machine translation. In the same paper, a translation system with discourse relations is described in a conceptual way. Later, Marcu *et al.* [131] noticed that the discourse structures (described in RST) of a single document in the source (Japanese) and target (English)

languages are different. Motivated by this observation, they tried to design some rules to transform a discourse structure between Japanese and English. Unfortunately, the structure transformation itself is a difficult problem. Therefore, it is impractical to apply this idea into a real translation system.

On the other hand, most recent work on discourse-based machine translation has largely focused on solutions to specific discourse phenomena: translating discourse connectives [136, 135], assigning appropriate morphology to pronouns [63, 111], maintaining lexical consistency [24], and employing the correct tense when it is not marked in the source language [208]. These various approaches are surveyed by Hardmeier [62], who finds that substantial BLEU score improvements are rare, even when the targeted phenomenon has been addressed with some success. Therefore, while a similar range of phenomena will be investigated in this section, a different approach is used to provide a general framework. In this framework, several issues discussed above could be addressed with a single model.

The rest of this section starts with a brief introduction of the greedy decoding idea, and then explains the base system on sentence-level translation. Finally, some preliminary results are presented with some discussion.

6.2.2 Greedy decoding

The greedy decoding starts with the computation of the hypothesis-pair scores from two adjacent source sentences. As illustrated in Figure 26, there are four source sentences, and each source sentence S_i has three hypotheses. Taking coDCLM as an example, it only considers the local context from the adjacent sentence pairs of a document. In this case, coDCLMis used to compute each hypothesis pair $(H_{i-1,j}, H_{i,k})$. For each hypothesis $H_{i,k}$, there are multiple context hypotheses from the translation of previous sentence S_{i-1}. Given a specific context $H_{i-1,j}$, different hypotheses $\{H_{i,k}\}_k$ may have different coDCLM scores. Let $d_{k,j}^{(i,i-1)}$ denote the coDCLM score of the

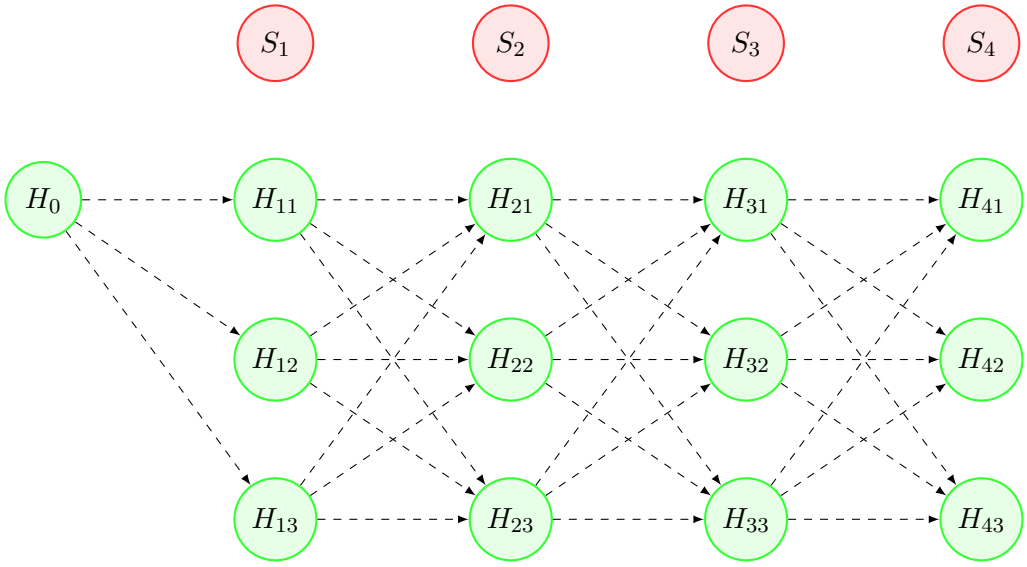


Figure 26: Every hypothesis pair $(H_{i-1,j}, H_{i,k})$ is used to compated a score for greedy decoding.

kth hypothesis of source sentence i conditional on the jth hypothesis of source sentence $i-1$. All these scores can be used as additional features for selecting a (local) optimal path from the trellis. Note that, there is a special case in this trellis — for the hypotheses of the first source sentence S_1, coDCLM assumes the default context (H_0). Then, all coDCLM scores of $\{H_{1,k}\}_k$ are computed conditional on H_0.

Figure 27 illustrates the procedure of greedy decoding on the trellis. A greedy decoder starts with H_0 to pick the first local optimal hypothesis of sentence S_1. Specifically, the decoder picks the optimal hypothesis based on the following log-linear model score

$$s_{k|i,i-1,j} \propto \exp(\sum_l w_l x^i_{k,l} + w_c d^{i,i-1}_{k,j}) \tag{57}$$

where $\{x^i_{k,l}\}$ are the features of the kth hypothesis given by a sentence-level translation system (Moses, in this case), $\{w_l\}$ are the corresponding Moses feature weights, and w_c is the feature weight of coDCLM score. (Training this log-linear model will be explained in subsection 6.2.4.) Therefore, the greedy decoding procedure not only depends on context, but the some additional features about the translated hypothesis

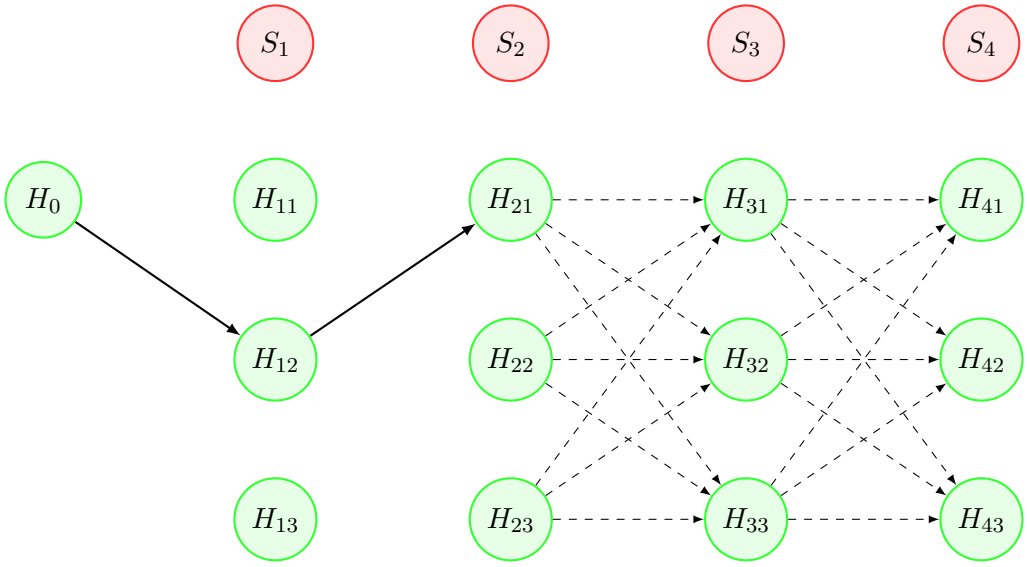


Figure 27: An illustration of the greedy decoding procedure.

itself. In Equation 57, the model assumes j in Equation 57 is fixed from the previous decoding step. Then, the greedy decoder simply pick the k that gives the largest $s_{k|i,i-1,j}$

$$k = \arg\max_{k'} s_{k'|i,i-1,j} \tag{58}$$

Figure 27 shows the middle of the greedy decoding procedure, where H_{12} and H_{21} are picked as the best hypotheses for source sentence S_1 and S_2 respectively. In next step, the decoder will use the coDCLM score conditional on H_{21} to find the optimal hypothesis for S_3. Eventually, an local optimal path is found from the trellis with the consideration of contextual information.

Two extensions can be considered with respect to this greedy decoding idea. First, given the trellis illustrated in Figure 26, it could be better to use the Viterbi decoding for finding the global optimal path. In addition, the second item in the right side of Equation 57 can be extended to include multiple context-based language model scores, such as ccDCLM, etc.

Table 14: TED Talks data statistics and reference 1-best/oracle BLEU scores for k-best reranKing (k=50) using a phrase-based MT system.

Data		es-en	fr-en	ja-en	zh-en
Training	Talks	1,038	1,032	737	997
	Sentences	140K	140K	92K	138K
	Words (en)	2,839K	2,830K	1,824K	2,781K
Development	Talks	20	20	20	20
	Sentences	1,478	1,478	1,478	1,478
	Words (en)	27K	27K	27K	27K
	1best BLEU	36.63	35.27	9.73	15.09
	Oracle BLEU	46.62	45.23	14.23	21.13
Test	Talks	20	20	20	20
	Sentences	1,616	1,616	1,616	1,616
	Words (en)	39K	39K	39K	39K
	1best BLEU	35.62	33.07	7.90	13.21
	Oracle BLEU	43.59	40.87	11.72	17.98

6.2.3 Data and Sentence-level translation system

The method proposed in this section is evaluated on the TED Talks corpus from the WIT^3 project [26][1]. TED Talks represent an interesting testbed for document-level translation because they consist of well-rehearsed monologues that are each coherent and focused document units. For preprocessing, the WIT^3 tools was used to for bitext extraction and dataset preparation. The TED translation corpus includes four datasets for translation into English: Spanish (es), French (fr), Japanese (ja), and Chinese (zh). The development set and test set for all four datasets come from the same 40 talks. Therefore, the target English side is the same. The data statistics are shown in Table 14.

For each dataset, a phrase-based machine translation system was trained using Moses [100]. From the training set, the phrase tables was established with the following features:

[1] `https://wit3.fbk.eu`

- GIZA++ word alignment, grow-diag-final and heuristic, and Good-Turing smoothing (5 features)
- lexical reordering tables with msd-bidirectional-fe option (7 features)
- SRILM 3gram language models with interpolated Kneser-Ney smoothing (1 feature)
- Word penalty (1 feature)

Therefore, the translation model has 14 features in total, which are also the Moses features used in Equation 57. To build a translation system, these feature weights are learned to optimize BLEU on the dev set. In the final step, the translation system generated 50-best lists with distinct English hypotheses on the development and test sets for greedy decoding.

6.2.4 Preliminary results

This preliminary experiment was conducted with the TED translation dataset mentioned before. To get an in-domain DCLMs, the training set of the TED data was used to train a set of language models, including RNNLM, coDCLM, and ccDCLM. Considering the size of TED training data, I also use the DCLM models trained the NANT dataset (refer to section 4.2). The similar model setup, including the input and hidden dimensions, from section 4.2 was used. The model with the best configuration was determined by the perplexity of the development set in the TED data. For each model, the score of each adjacent hypothesis pair was computed as the context-sensitive features for greedy decoding. Note that, the way of computing DCLM scores is using the coDCLM, since RNNLMis context-independent and a better way to use ccDCLM is to use all previous sentences as context.

Recall that, the second item in Equation 57 can be extended to include multiple DCLM scores. The experiment considers three model variations by adding DCLM

Table 15: BLEU scores of greedy decoding with different document context language models on four translation datasets.

	zh-en		fr-en		es-en		ja-en	
	Dev	Test	Dev	Test	Dev	Test	Dev	Test
1best	15.09	13.21	35.27	33.07	36.63	35.62	9.73	7.90
Model 1	15.71	13.55	36.28	33.77	38.00	36.41	10.05	8.18
Model 2	15.70	13.73	36.29	34.03	38.08	36.43	10.00	8.21
Model 3	15.62	13.84	36.28	33.97	38.30	36.59	10.01	8.15

scores in an incremental way.

- Model 1: RNNLMtrained on the TED data and the NANT data respectively. Both of them do not include any contextual information. The purpose of using this model is to further improve the BLEU with better sentence-level language models.

- Model 2: Besides model 1, also incorporate CODCLM trained on the TED data and the NANT data respectively. Together, four score features are used together Moses features for decoding. The performance based on this model show the BLEU improvement with contextual information.

- Model 3: Besides model 2, also include two CCDCLM features. The expectation is to show possible further improvement with additional contextual information.

The weights of DCLM features (together with other Moses features weights in Equation 57) was obtained by applying the Minimum Error Rate Training (MERT) on the development set.

Table 15 shows the BLEU scores of greedy decoding on four different translation datasets. Overall, adding DCLM features improves the BLEU score on document-level translation. The biggest improvement across four source languages is up to 0.9 BLEU score. More important, the BLEU improvement is independent with the source language and the baseline BLEU score. However, for each source language, if

the best is picked with respect to the best performance on the development set, some results are not consistent or unexpected. For example, in Chinese (zh) to English (en), the best performance is given by model 1 (13.55 BLEU score), where the BLEU improvement on the test set is lower than the best BLEU we can obtain on the test set (model 3, 13.84 BLEU score). Additionally, model 1 does not any contextual information. Therefore, we cannot draw any conclusion with respect to contextual information for machine translation.

In future work, the performance is expected to improve with some additional machine translation techniques. For example, replace greedy decoding with Viterbi decoding or directly incorporate contextual information into translation.

CHAPTER VII

CONCLUSION

Recall that the central thesis of this dissertation is that "*semantic representation learning can help discourse processing*". In this dissertation, I verify this claim from the following three scenarios:

- With supervision information from discourse annotation, representation learning can capture the semantic information either from word-level or from sentence-level. Then, the distributed representation driven by discourse information further boost the discourse processing performance on both RST-style and PDTB-style parsing tasks.

- Semantic representation learning can be performed in the multi-task learning framework. To be specific, in my work, the model learns the distributed representation of sentences together with word embeddings, by given the supervision information from discourse annotation. Then, a single model can either be used for discourse processing or for language modeling.

- My work also shows that discourse processing can benefit from representation learning even without direct supervision information. There exists some ways to leverage the supervision information from some "easier" cases to the "harder" cases, as explained in chapter 5.

However, this is not the end of the story. There are lots possible directions we can go in future to make this thesis even more solid and make the performance of our discourse processing system even better. For example:

Supervised representation learning Future work could consider joint models of discourse structure and coreference, and consideration of coreference across the entire document. In the longer term, we hope to induce and exploit representations of other discourse elements, such as event coreference and shallow semantics.

Discourse-driven language modeling Future work on DCLMs could include testing the applicability of these models to downstream applications such as summarization and translation. Future work on DrLM will investigate the possibility of learning from partially-labeled training data, which would have at least two potential advantages. First, it would enable the model to scale up to the large datasets needed for competitive language modeling. Second, by training on more data, the resulting vector representations might support even more accurate discourse relation prediction.

Discourse processing with distant supervision Future work could explore the combination of this approach with more sophisticated techniques for instance selection [169] and feature selection [156, 13], while also tackling the more difficult problems of multi-class relation classification and fine-grained level-2 discourse relations. conclusion for reference

REFERENCES

[1] Asher, N. and Lascarides, A., *Logics of conversation*. Cambridge University Press, 2003. 2.1.1

[2] Bach, N. X., Minh, N. L., and Shimazu, A., "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*, pp. 160–168, Association for Computational Linguistics, 2012. 2.2.1, 3.1.3

[3] Bahdanau, D., Cho, K., and Bengio, Y., "Neural machine translation by jointly learning to align and translate," in *International Conference on Learning Representations*, 2015. 4.2.2

[4] Baroni, M., Bernardi, R., and Zamparelli, R., "Frege in space: A program of compositional distributional semantics," *LiLT (Linguistic Issues in Language Technology)*, vol. 9, 2014. 3.2

[5] Barzilay, R. and Lapata, M., "Modeling local coherence: An entity-based approach," *Computational Linguistics*, vol. 34, no. 1, pp. 1–34, 2008. 4.2.3.2, 6.2

[6] Ben-David, S., Blitzer, J., Crammer, K., Pereira, F., and others, "Analysis of representations for domain adaptation," *Advances in neural information processing systems*, vol. 19, 2007. 5.1

[7] Bengio, Y., "Practical recommendations for gradient-based training of deep architectures," in *Neural Networks: Tricks of the Trade*, pp. 437–478, Springer, 2012. 3.2.2, 4.3.5

[8] Bengio, Y., Courville, A., and Vincent, P., "Representation Learning: A Review and New Perspectives," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1798–1828, 2013. 2.3.1, 2.3.1

[9] Bengio, Y., Ducharme, R., Vincent, P., and Janvin, C., "A neural probabilistic language model," *The Journal of Machine Learning Research*, vol. 3, pp. 1137–1155, 2003. 4.1

[10] Bengio, Y., Simard, P., and Frasconi, P., "Learning long-term dependencies with gradient descent is difficult," *Neural Networks, IEEE Transactions on*, vol. 5, no. 2, pp. 157–166, 1994. 3.2.2, 4

[11] Berg, G., "Learning recursive phrase structure: Combining the strengths of PDP and X-bar syntax," tech. rep., State University of New York at Albany, 1991. TR 91-5. 2.4.3

[12] Bhatia, P., Ji, Y., and Eisenstein, J., "Better document-level sentiment analysis from rst discourse parsing," in *EMNLP*, 2015. 6

[13] Biran, O. and McKeown, K., "Aggregated word pair features for implicit discourse relation disambiguation," in *ACL*, 2013. 2.2.2, 2.4.1, 3.2.3, 3.2.3.2, 5, 4.3.5, 5.1, 5.2.3, 7

[14] Bishop, C. M., *Neural networks for pattern recognition*. Oxford university press, 1995. 1.3

[15] Bishop, C. M., *Pattern recognition and machine learning*. springer, 2006. 1.1, 1.3.2, 2.3, 2, 6.2

[16] Blacoe, W. and Lapata, M., "A comparison of vector-based representations for semantic composition," in *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pp. 546–556, Association for Computational Linguistics, 2012. 1.3, 3.2.3.1

[17] Blei, D. M., Ng, A. Y., and Jordan, M. I., "Latent dirichlet allocation," *the Journal of machine Learning research*, vol. 3, pp. 993–1022, 2003. 2.4.2

[18] Blitzer, J., McDonald, R., and Pereira, F., "Domain adaptation with structural correspondence learning," in *Proceedings of the 2006 conference on empirical methods in natural language processing*, pp. 120–128, Association for Computational Linguistics, 2006. 5.1, 5.2.1

[19] Bottou, L., "Online Algorithms and Stochastic Approximations," in *Online Learning and Neural Networks* (Saad, D., ed.), Cambridge, UK: Cambridge University Press, 1998. revised, oct 2012. 3.2.2

[20] Bottou, L., "Stochastic gradient descent tricks," in *Neural Networks: Tricks of the Trade*, pp. 421–436, Springer, 2012. 2.3.1

[21] Burstein, J., Tetreault, J., and Chodorow, M., "Holistic Discourse Coherence Annotation for Noisy Essay Writing," *Dialogue & Discourse*, vol. 4, no. 2, pp. 34–52, 2013. 1

[22] Carlson, L. and Marcu, D., "Discourse tagging reference manual," *ISI Technical Report ISI-TR-545*, vol. 54, 2001. 3.1.3

[23] Carlson, L., Marcu, D., and Okurowski, M. E., "Building a Discourse-tagged Corpus in the Framework of Rhetorical Structure Theory," in *Proceedings of Second SIGdial Workshop on Discourse and Dialogue*, 2001. 2.1.1, 3.1, 3.1.3

[24] Carpuat, M. and Simard, M., "The trouble with smt consistency," in *Proceedings of the Seventh Workshop on Statistical Machine Translation*, pp. 442–449, Association for Computational Linguistics, 2012. 6.2.1

[25] Caruana, R., "Multitask learning," *Machine learning*, vol. 28, no. 1, pp. 41–75, 1997. 4.3.4

[26] Cettolo, M. and Girardi, C., "Wit3: Web inventory of transcribed and translated talks," in *Proceedings of the 16th EAMT Conference*, 2012. 6.2.3

[27] Chen, M., Xu, Z., Weinberger, K., and Sha, F., "Marginalized Denoising Autoencoders for Domain Adaptation," in *Proceedings of the 29th International Conference on Machine Learning*, pp. 767–774, 2012. 5.1, 5.2.1

[28] Chenlo, J. M., Hogenboom, A., and Losada, D. E., "Rhetorical structure theory for polarity estimation: An experimental study," *Data & Knowledge Engineering*, vol. 94, pp. 135–147, 2014. 6.1.1

[29] Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y., "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014. 4.1

[30] Chung, J., Gulcehre, C., Cho, K., and Bengio, Y., "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv preprint arXiv:1412.3555*, 2014. 4.2.1

[31] Chung, J., Kastner, K., Dinh, L., Goel, K., Courville, A. C., and Bengio, Y., "A recurrent latent variable model for sequential data," in *Advances in neural information processing systems*, pp. 2962–2970, 2015. 4.3, 4.3.2

[32] Cohen, S. B., Stratos, K., Collins, M., Foster, D. P., and Ungar, L., "Spectral learning of latent-variable pcfgs: Algorithms and sample complexity," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 2399–2449, 2014. 3.2.1

[33] Collins, M. and Roark, B., "Incremental parsing with the perceptron algorithm," in *Proceedings of ACL*, p. 111, Association for Computational Linguistics, 2004. 3.1.1

[34] Collobert, R. and Weston, J., "A unified architecture for natural language processing: Deep neural networks with multitask learning," in *Proceedings of the 25th international conference on Machine learning*, pp. 160–167, ACM, 2008. 3.1.3

[35] Crammer, K. and Singer, Y., "On the algorithmic implementation of multiclass kernel-based vector machines," *The Journal of Machine Learning Research*, vol. 2, pp. 265–292, 2002. 3.1.2

[36] Cristea, D., Ide, N., and Romary, L., "Veins theory: A model of global discourse cohesion and coherence," in *Proceedings of the 17th international conference on Computational linguistics-Volume 1*, pp. 281–285, Association for Computational Linguistics, 1998. 3.2

[37] Davison, A. C. and Hinkley, D. V., *Bootstrap methods and their application*, vol. 1. Cambridge university press, 1997. 4.2.3.2

[38] de Freitas, J. F., Niranjan, M., Gee, A. H., and Doucet, A., "Sequential monte carlo methods to train neural network models," *Neural computation*, vol. 12, no. 4, pp. 955–993, 2000. 4.3.2

[39] De Mulder, W., Bethard, S., and Moens, M.-F., "A survey on the application of recurrent neural networks to statistical language modeling," *Computer Speech & Language*, vol. 30, no. 1, pp. 61–98, 2015. 4.1

[40] Denil, M., Demiraj, A., Kalchbrenner, N., Blunsom, P., and de Freitas, N., "Modelling, visualising and summarising documents with a single convolutional neural network," *arXiv preprint arXiv:1406.3830*, 2014. 4.1

[41] Dinu, G. and Lapata, M., "Measuring distributional similarity in context," in *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pp. 1162–1172, Association for Computational Linguistics, 2010. 3.1.3

[42] Dolan, B., Quirk, C., and Brockett, C., "Unsupervised Construction of Large Paraphrase Corpora: Exploiting Massively Parallel News Sources," in *COLING*, 2004. 3

[43] Duchi, J., Hazan, E., and Singer, Y., "Adaptive subgradient methods for online learning and stochastic optimization," *The Journal of Machine Learning Research*, vol. 12, pp. 2121–2159, 2011. 3.2.2, 4.2.3, 4.3.5

[44] Dumais, S. T., "Latent semantic analysis," *Annual review of information science and technology*, vol. 38, no. 1, pp. 188–230, 2004. 2.4.2

[45] Durrett, G. and Klein, D., "Easy Victories and Uphill Battles in Coreference Resolution," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, (Seattle, Washington), Association for Computational Linguistics, October 2013. 3.2.2

[46] Eisenstein, J., "A technical introduction to statistical natural language processing," 2016. 2.3, 3

[47] Fan, R.-E., Chang, K.-W., Hsieh, C.-J., Wang, X.-R., and Lin, C.-J., "Liblinear: A library for large linear classification," *The Journal of Machine Learning Research*, vol. 9, pp. 1871–1874, 2008. 5.2.3

[48] Feldman, R., "Techniques and applications for sentiment analysis," *Communications of the ACM*, vol. 56, no. 4, pp. 82–89, 2013. 6.1

[49] Feng, V. W. and Hirst, G., "Text-level discourse parsing with rich linguistic features," in *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pp. 60–68, Association for Computational Linguistics, 2012. 3.1, 3.1.3, 3.1.3

[50] Feng, V. W. and Hirst, G., "A Linear-Time Bottom-Up Discourse Parser with Constraints and Post-Editing," in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, (Baltimore, Maryland), pp. 511–521, Association for Computational Linguistics, June 2014. 2.4.1, 2.4.1

[51] Ferrucci, D., Brown, E., Chu-Carroll, J., Fan, J., Gondek, D., Kalyanpur, A. A., Lally, A., Murdock, J. W., Nyberg, E., Prager, J., and others, "Building Watson: An overview of the DeepQA project," *AI magazine*, vol. 31, no. 3, pp. 59–79, 2010. 1

[52] Forbes-Riley, K., Webber, B., and Joshi, A., "Computing discourse semantics: The predicate-argument semantics of discourse connectives in D-LTAG," *Journal of Semantics*, vol. 23, no. 1, pp. 55–106, 2006. 2.1.2, 3.2

[53] Ghosh, S., Riccardi, G., and Johansson, R., "Global features for shallow discourse parsing," in *Proceedings of the 13th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pp. 150–159, Association for Computational Linguistics, 2012. 2.4.1

[54] Glorot, X. and Bengio, Y., "Understanding the difficulty of training deep feedforward neural networks," in *International conference on artificial intelligence and statistics*, pp. 249–256, 2010. 4.2.3

[55] Glorot, X., Bordes, A., and Bengio, Y., "Domain adaptation for large-scale sentiment classification: A deep learning approach," in *Proceedings of the 28th International Conference on Machine Learning*, pp. 513–520, 2011. 5.2.1

[56] Godfrey, J. J., Holliman, E. C., and McDaniel, J., "Switchboard: Telephone speech corpus for research and development," in *Acoustics, Speech, and Signal Processing, 1992. ICASSP-92., 1992 IEEE International Conference on*, vol. 1, pp. 517–520, IEEE, 1992. 4.3.5

[57] Goller, C. and Kuchler, A., "Learning task-dependent distributed representations by backpropagation through structure," in *Neural Networks, 1996., IEEE International Conference on*, vol. 1, pp. 347–352, IEEE, 1996. 2.4.3, 3.2.2, 6.1.3

[58] Grice, H. P., "Logic and Conversation," in *Syntax and Semantics Volume 3: Speech Acts* (Cole, P. and Morgan, J. L., eds.), pp. 41–58, Academic Press, 1975. 5

[59] Grosz, B. J. and Sidner, C. L., "Attention, intentions, and the structure of discourse," *Computational linguistics*, vol. 12, no. 3, pp. 175–204, 1986. 2.1

[60] Guo, W. and Diab, M., "Modeling sentences in the latent space," in *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pp. 864–872, Association for Computational Linguistics, 2012. 3

[61] Halpern, J. Y., "Reasoning about knowledge: an overview," in *Proceedings of the 1986 Conference on Theoretical aspects of reasoning about knowledge*, pp. 1–17, Morgan Kaufmann Publishers Inc., 1986. 1

[62] Hardmeier, C., *Discourse in statistical machine translation*. Acta Universitatis Upsaliensis, 2014. 6.2.1

[63] Hardmeier, C., Tiedemann, J., and Nivre, J., "Latent anaphora resolution for cross-lingual pronoun prediction," in *EMNLP 2013; Conference on Empirical Methods in Natural Language Processing; 18-21 October 2013; Seattle, WA, USA*, pp. 380–391, Association for Computational Linguistics, 2013. 6.2.1

[64] Harris, Z. S., "Distributional Structure," *Word*, 1954. 1.2, 1.3, 2.4.2

[65] Heerschop, B., Goossen, F., Hogenboom, A., Frasincar, F., Kaymak, U., and de Jong, F., "Polarity analysis of texts using discourse structure," in *Proceedings of the 20th ACM international conference on Information and knowledge management*, pp. 1061–1070, ACM, 2011. 6.1.1

[66] Hernault, H., Prendinger, H., duVerle, D. A., and Ishizuka, M., "HILDA: A Discourse Parser Using Support Vector Machine Classification," *Dialogue and Discourse*, vol. 1, no. 3, pp. 1–33, 2010. (document), 2.2.1, 2.2.1, 2.4.1, 2, 3.1.3

[67] Hinton, G., "Learning distributed representations of concepts," in *Proceedings of the Eighth Annual Conference of the Cognitive Science Society*, pp. 1–12, 2012. 2.4.2

[68] Hinton, G., McClelland, J., and Rumelhart, D., "Distributed representations," *Parallel Distributed Processing: Explorations in the micro-structure of cognition*, 1984. 1.2, 2.4.2

[69] Hirao, T., Yoshida, Y., Nishino, M., Yasuda, N., and Nagata, M., "Single-Document Summarization as a Tree Knapsack Problem.," in *EMNLP*, pp. 1515–1520, 2013. 6.1.2, 6.1.2

[70] Hobbs, J. R., "Coherence and coreference," *Cognitive science*, vol. 3, no. 1, pp. 67–90, 1979. 1, 2.1, 2.1.1, 2.4

[71] Hobbs, J. R., "On the coherence and structure of discourse," 1985. 1, 2.1

[72] Hobbs, J. R., "Intention, information, and structure in discourse: A first draft," in *Burning Issues in Discourse, NATO Advanced Research Workshop*, pp. 41–66, 1993. 2.1

[73] Hochreiter, S. and Schmidhuber, J., "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997. 4.2.1

[74] Hogenboom, A., Frasincar, F., de Jong, F., and Kaymak, U., "Using rhetorical structure in sentiment analysis," *Communications of the ACM*, vol. 58, no. 7, pp. 69–77, 2015. 6.1.1

[75] Hovy, E. H., "Automated discourse generation using discourse structure relations," *Artificial intelligence*, vol. 63, no. 1, pp. 341–385, 1993. 1

[76] Irsoy, O. and Cardie, C., "Bidirectional recursive neural networks for token-level labeling with structure," in *NIPS 2013 Workshop on Deep Learning*, 2013. 2.4.3

[77] Jansen, P., Surdeanu, M., and Clark, P., "Discourse Complements Lexical Semantics for Non-factoid Answer Reranking," in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, (Baltimore, Maryland), pp. 977–986, Association for Computational Linguistics, June 2014. 1

[78] Ji, Y., Cohn, T., Kong, L., Dyer, C., and Eisenstein, J., "Document Context Language Models," in *ICLR (Workshop track)*, 2016. 1

[79] Ji, Y., Cohn, T., Kong, L., Dyer, C., and Eisenstein, J., "Document Context Language Models," in *ICLR (Workshop track)*, 2016. 4

[80] Ji, Y. and Eisenstein, J., "Discriminative Improvements to Distributional Sentence Similarity," in *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, (Seattle, Washington, USA), pp. 891–896, Association for Computational Linguistics, October 2013. 3

[81] Ji, Y. and Eisenstein, J., "Representation learning for document-level discourse parsing," in *ACL*, 2014. 1.6, 2.3.1, 2.4.2, 3, 3.2.2, 6.1.3

[82] Ji, Y. and Eisenstein, J., "One Vector is Not Enough: Entity-Augmented Distributed Semantics for Discourse Relations," *Transactions of the Association of Computational Linguistics*, vol. 3, pp. 329–344, 2015. 2.4.3, 3, 4.3, 4.3.5, 9

[83] Ji, Y., Haffari, G., and Eisenstein, J., "A latent variable recurrent neural network for discourse-driven language models," in *NAACL-HLT*, 2016. 4

[84] Ji, Y., Zhang, G., and Eisenstein, J., "Closing the Gap: Domain Adaptation from Explicit to Implicit Discourse Relations," in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, (Lisbon, Portugal), pp. 2219–2224, Association for Computational Linguistics, September 2015. 5

[85] Jiang, J. and Zhai, C., "Instance Weighting for Domain Adaptation in NLP," in *Proceedings of ACL*, 2007. 5.1, 5.2.2

[86] Joachims, T., *Learning to classify text using support vector machines: Methods, theory and algorithms*. Kluwer Academic Publishers, 2002. 1

[87] Johnstone, B., *Discourse analysis*. Blackwell Malden, MA, 2008. 2.1

[88] Joshi, M., Cohen, W. W., Dredze, M., and Rosé, C. P., "Multi-domain learning: when do domains matter?," in *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pp. 1302–1312, Association for Computational Linguistics, 2012. 5.1, 5.2.2

[89] Joty, S., Carenini, G., and Ng, R., "A Novel Discriminative Framework for Sentence-Level Discourse Analysis," in *EMNLP-CoNLL*, 2012. 2.2.1

[90] Joty, S. R., Carenini, G., Ng, R. T., and Mehdad, Y., "Combining intra- and multi-sentential rhetorical parsing for document-level discourse analysis.," in *ACL (1)*, pp. 486–496, 2013. (document), 2.2.1, 2.4.1, 2.4.1, 3.1, 3.1.2, 2, 3.1.3, 3.1.3

[91] Jurafsky, D. and Martin, J. H., *Speech & language processing*. Pearson Education India, 2000. 2.2.1, 4

[92] Jurafsky, D., Shriberg, E., and Biasca, D., "Switchboard SWBD-DAMSL shallow-discourse-function annotation coders manual," *Institute of Cognitive Science Technical Report*, pp. 97–102, 1997. 4.3.1

[93] Kalchbrenner, N. and Blunsom, P., "Recurrent Convolutional Neural Networks for Discourse Compositionality," *Proceedings of the 2013 Workshop on Continuous Vector Space Models and their Compositionality*, 2013. 4.3, 4.3.5, 10

[94] Kiela, D., Hill, F., and Clark, S., "Specializing word embeddings for similarity or relatedness," in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, September 2015. 2.4.2

[95] Klein, D. and Manning, C. D., "Accurate unlexicalized parsing," in *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*, pp. 423–430, Association for Computational Linguistics, 2003. 3.2.2

[96] Knight, K., "Decoding complexity in word-replacement translation models," *Computational Linguistics*, vol. 25, no. 4, pp. 607–615, 1999. 6.2

[97] Knott, A. and Dale, R., "Using linguistic phenomena to motivate a set of coherence relations," *Discourse processes*, vol. 18, no. 1, pp. 35–62, 1994. 2.4.1

[98] Knott, A., Sanders, T., and Oberlander, J., "Levels of representation in discourse relations," *Cognitive Linguistics*, vol. 12, no. 3, pp. 197–210, 2001. 2.1.1

[99] Koehn, P., *Statistical machine translation*. Cambridge University Press, 2009. 1.5.2, 4

[100] Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., and others, "Moses: Open source toolkit for statistical machine translation," in *Proceedings of the 45th annual meeting of the ACL on interactive poster and demonstration sessions*, pp. 177–180, Association for Computational Linguistics, 2007. 6.2.3

[101] Koo, T., Carreras, X., and Collins, M., "Simple Semi-supervised Dependency Parsing," in *Proceedings of ACL-HLT*, (Columbus, Ohio), pp. 595–603, Association for Computational Linguistics, June 2008. 3.1.1

[102] Krestel, R., Bergler, S., Witte, R., and others, "Minding the source: Automatic tagging of reported speech in newspaper articles," *Reporter*, vol. 1, no. 5, p. 4, 2008. 3.1.3

[103] Kübler, S., McDonald, R., and Nivre, J., *Dependency parsing*. Morgan & Claypool Publishers, 2009. 6.1.2

[104] Lafferty, J., McCallum, A., and Pereira, F. C., "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," in *Proceedings of the 18th International Conference on Machine Learning*, 2001. 2.2.1

[105] Lan, M., Xu, Y., and Niu, Z., "Leveraging Synthetic Discourse Data via Multi-task Learning for Implicit Discourse Relation Recognition," in *Proceedings of ACL*, (Sofia, Bulgaria), pp. 476–485, Association for Computational Linguistics, August 2013. 2.2.2, 2.4.1, 5.1

[106] Lari, K. and Young, S. J., "The estimation of stochastic context-free grammars using the inside-outside algorithm," *Computer speech & language*, vol. 4, no. 1, pp. 35–56, 1990. 3.2.1

[107] Lascarides, A. and Asher, N., "Temporal interpretation, discourse relations and commonsense entailment," *Linguistics and philosophy*, vol. 16, no. 5, pp. 437–493, 1993. 2.1

[108] Lazaridou, A., Titov, I., and Sporleder, C., "A bayesian model for joint unsupervised induction of sentiment, aspect and discourse representations.," in *ACL*, pp. 1630–1639, 2013. 6.1.1

[109] Le, P. and Zuidema, W., "The inside-outside recursive neural network model for dependency parsing," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 729–739, 2014. 2.4.3

[110] Le, Q. V. and Mikolov, T., "Distributed representations of sentences and documents," *arXiv preprint arXiv:1405.4053*, 2014. 4, 4.1

[111] Le Nagard, R. and Koehn, P., "Aiding pronoun translation with co-reference resolution," in *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*, pp. 252–261, Association for Computational Linguistics, 2010. 6.2.1

[112] LeCun, Y. A., Bottou, L., Orr, G. B., and Müller, K.-R., "Efficient backprop," in *Neural networks: Tricks of the trade*, pp. 9–48, Springer, 2012. 3.2.2

[113] Levin, B., "Aspect, lexical semantic representation, and argument expression," in *Proceedings of the 26th annual meeting of the Berkeley Linguistics Society*, pp. 413–429, Citeseer, 2000. 2.4.1

[114] Li, J. and Hovy, E. H., "A model of coherence based on distributed sentence representation.," in *EMNLP*, pp. 2039–2048, 2014. 4.1, 4.2.3.2

[115] Li, J., Li, R., and Hovy, E., "Recursive Deep Models for Discourse Parsing," in *EMNLP*, October 2014. 1.3.1

[116] Li, J., Luong, M.-T., and Jurafsky, D., "A hierarchical neural autoencoder for paragraphs and documents," *arXiv preprint arXiv:1506.01057*, 2015. 4.1, 4.2.3.2

[117] Lin, R., Liu, S., Yang, M., Li, M., Zhou, M., and Li, S., "Hierarchical recurrent neural network for document modeling," in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 899–907, 2015. 4, 4.1, 7, 4.2.3, 8, 4.2.3.2

[118] Lin, Z., Kan, M.-Y., and Ng, H. T., "Recognizing implicit discourse relations in the Penn Discourse Treebank," in *EMNLP*, 2009. (document), 2.2.2, 2.4.1, 2.4.1, 2.4.1, 3.2.1, 3.2.2, 3.2.3, 3.2.3.1, 4, 4.3.5, 5.2.3

[119] Lin, Z., Ng, H. T., and Kan, M.-Y., "A PDTB-styled end-to-end discourse parser," *Natural Language Engineering*, pp. 1–34, 2014. 2.2.2, 2.2.2, 2.3.1, 3.2.3

[120] Liu, B., "Sentiment analysis and opinion mining," *Synthesis lectures on human language technologies*, vol. 5, no. 1, pp. 1–167, 2012. 6.1

[121] Louis, A., Joshi, A., and Nenkova, A., "Discourse indicators for content selection in summarization," in *Proceedings of the 11th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pp. 147–156, Association for Computational Linguistics, 2010. 1

[122] Louis, A., Joshi, A., Prasad, R., and Nenkova, A., "Using entity features to classify implicit discourse relations," in *Proceedings of the SIGDIAL*, (Tokyo, Japan), pp. 59–62, Association for Computational Linguistics, September 2010. 2.4.1, 3.2

[123] Maas, A. L., Daly, R. E., Pham, P. T., Huang, D., Ng, A. Y., and Potts, C., "Learning word vectors for sentiment analysis," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pp. 142–150, Association for Computational Linguistics, 2011. 2.4.2

[124] Mann, W. and Thompson, S., "Rhetorical Structure Theory: Toward a Functional Theory of Text Organization," *Text*, vol. 8, no. 3, pp. 243–281, 1988. 2.1.1

[125] Mann, W. C., "Discourse Structures for Text Generation," in *Proceedings of the 10th International Conference on Computational Linguistics and 22nd annual meeting on Association for Computational Linguistics*, pp. 367–375, Association for Computational Linguistics, 1984. 2.1.1

[126] Manning, C. D., Raghavan, P., Schütze, H., and others, *Introduction to information retrieval*, vol. 1. Cambridge university press Cambridge, 2008. 4

[127] Manning, C. D., Surdeanu, M., Bauer, J., Finkel, J. R., Bethard, S., and McClosky, D., "The stanford corenlp natural language processing toolkit.," in *ACL (System Demonstrations)*, pp. 55–60, 2014. 5.2.3

[128] Marcu, D., "Building Up Rhetorical Structure Trees," in *Proceedings of AAAI*, 1996. 3.1.1

[129] Marcu, D., "A Decision-Based Approach to Rhetorical Parsing," in *Proceedings of ACL*, (College Park, Maryland, USA), pp. 365–372, Association for Computational Linguistics, June 1999. 2.2.1, 2.4.1, 2.4.1, 2.4.1, 3.1, 3.1.1

[130] Marcu, D., *The Theory and Practice of Discourse Parsing and Summarization*. MIT Press, 2000. 3.1.3

[131] Marcu, D., Carlson, L., and Watanabe, M., "The automatic translation of discourse structures," in *Proceedings of the 1st North American chapter of the Association for Computational Linguistics conference*, pp. 9–17, Association for Computational Linguistics, 2000. 6.2.1

[132] Marcu, D. and Echihabi, A., "An unsupervised approach to recognizing discourse relations," in *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pp. 368–375, Association for Computational Linguistics, 2002. 5.1

[133] Marcus, M. P., Marcinkiewicz, M. A., and Santorini, B., "Building a large annotated corpus of english: The penn treebank," *Computational linguistics*, vol. 19, no. 2, pp. 313–330, 1993. 4.2.3

[134] McClosky, D., Charniak, E., and Johnson, M., "Bllip north american news text, complete," *Linguistic Data Consortium*, 2008. 4.2.3

[135] Meyer, T., Hajlaoui, N., and Popescu-Belis, A., "Disambiguating discourse connectives for statistical machine translation," *Audio, Speech, and Language Processing, IEEE/ACM Transactions on*, vol. 23, no. 7, pp. 1184–1197, 2015. 6.2.1

[136] Meyer, T., Popescu-Belis, A., Hajlaoui, N., and Gesmundo, A., "Machine translation of labeled discourse connectives," in *Proceedings of the Tenth Biennial Conference of the Association for Machine Translation in the Americas (AMTA)*, no. EPFL-CONF-192524, 2012. 6.2.1

[137] Mikolov, T., Karafiát, M., Burget, L., Cernockỳ, J., and Khudanpur, S., "Recurrent neural network based language model.," in *INTERSPEECH 2010, 11th Annual Conference of the International Speech Communication Association, Makuhari, Chiba, Japan, September 26-30, 2010*, pp. 1045–1048, 2010. 1.3.2, 4, 4.1, 4.2.1, 4.2.3, 4.2.3, 7, 4.3, 4.3.5

[138] Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J., "Distributed representations of words and phrases and their compositionality," in *Advances in Neural Information Processing Systems*, pp. 3111–3119, 2013. 2.4.2

[139] Mikolov, T., Yih, W.-t., and Zweig, G., "Linguistic Regularities in Continuous Space Word Representations," in *NAACL-HLT*, (Atlanta, Georgia), pp. 746–751, Association for Computational Linguistics, June 2013. 1.3, 3.2.2

[140] Mikolov, T. and Zweig, G., "Context dependent recurrent neural network language model.," in *SLT*, pp. 234–239, 2012. 4, 4.1

[141] Milajevs, D. and Purver, M., "Investigating the Contribution of Distributional Semantic Information for Dialogue Act Classification," in *Proceedings of the 2nd Workshop on Continuous Vector Space Models and their Compositionality (CVSC)@ EACL*, pp. 40–47, 2014. 4.3.5

[142] Miller, S., Guinness, J., and Zamanian, A., "Name Tagging with Word Clusters and Discriminative Training," in *HLT-NAACL* (Susan Dumais, D. M. and Roukos, S., eds.), (Boston, Massachusetts, USA), pp. 337–342, Association for Computational Linguistics, May 2 - May 7 2004. 3.1.1

[143] Miltsakaki, E. and Kukich, K., "Evaluation of text coherence for electronic essay scoring systems," *Natural Language Engineering*, vol. 10, no. 01, pp. 25–55, 2004. 1

[144] Mitchell, J. and Lapata, M., "Composition in distributional models of semantics," *Cognitive science*, vol. 34, no. 8, pp. 1388–1429, 2010. 1.3, 3.1.3

[145] Mitkov, R., "How could rhetorical relations be used in machine translation?(And at least two open questions)," in *Proceedings of the ACL Workshop on Intentionality and Structure in Discourse Relations*, pp. 86–89, 1993. 6.2.1

[146] Mooney, R., "Semantic Parsing: Past, Present, and Future," 2014. Invited talk on ACL 2014 workshop on semantic parsing. 1

[147] Moore, J. D. and Wiemer-Hastings, P., "Discourse in computational linguistics and artificial intelligence," *Handbook of discourse processes*, pp. 439–486, 2003. 2.1

[148] Murphy, K. P., *Machine Learning: A Probabilistic Perspective*. The MIT Press, 2012. 2.3, 2.3.1

[149] Nelakanti, A. K., Archambeau, C., Mairal, J., Bach, F., and Bouchard, G., "Structured Penalties for Log-Linear Language Models," in *EMNLP*, (Seattle, Washington, USA), pp. 233–243, Association for Computational Linguistics, October 2013. 3.1.2

[150] Ngiam, J., Coates, A., Lahiri, A., Prochnow, B., Le, Q. V., and Ng, A. Y., "On optimization methods for deep learning," in *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pp. 265–272, 2011. 3.1.2

[151] Nivre, J., Hall, J., Nilsson, J., Chanev, A., Eryigit, G., Kübler, S., Marinov, S., and Marsi, E., "MaltParser: A language-independent system for data-driven dependency parsing," *Natural Language Engineering*, vol. 13, no. 2, pp. 95–135, 2007. 3.1.2

[152] Ovchinnikova, E., Montazeri, N., Alexandrov, T., Hobbs, J. R., McCord, M. C., and Mulkar-Mehta, R., "Abductive reasoning with a large knowledge base for discourse processing," in *Computing Meaning*, pp. 107–127, Springer, 2014. 2.4

[153] Pang, B. and Lee, L., "A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts," in *Proceedings of the 42nd annual meeting on Association for Computational Linguistics*, p. 271, Association for Computational Linguistics, 2004. (document), 6.1.1, 13

[154] Pang, B. and Lee, L., "Opinion mining and sentiment analysis," *Foundations and trends in information retrieval*, vol. 2, no. 1-2, pp. 1–135, 2008. 6.1

[155] Pang, B., Lee, L., and Vaithyanathan, S., "Thumbs up?: sentiment classification using machine learning techniques," in *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*, pp. 79–86, Association for Computational Linguistics, 2002. 1.5.1

[156] Park, J. and Cardie, C., "Improving Implicit Discourse Relation Recognition Through Feature Set Optimization," in *Proceedings of the 13th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, (Seoul, South Korea), pp. 108–112, Association for Computational Linguistics, July 2012. 2.2.2, 2.4.1, 3.2.3, 3.2.3.2, 5, 3.2.3.2, 4.3.5, 5.2.3, 7

[157] Partee, B., "Lexical semantics and compositionality," *An invitation to cognitive science: Language*, vol. 1, pp. 311–360, 1995. 1.3.1

[158] Pascanu, R., Mikolov, T., and Bengio, Y., "On the difficulty of training recurrent neural networks," *arXiv preprint arXiv:1211.5063*, 2012. 3.2.1, 3.2.2, 4.2.3, 4.3.5

[159] Paulus, R., Socher, R., and Manning, C. D., "Global Belief Recursive Neural Networks," in *Advances in Neural Information Processing Systems*, pp. 2888–2896, 2014. 2.4.3, 2.4.3

[160] Pennington, J., Socher, R., and Manning, C. D., "Glove: Global vectors for word representation," in *EMNLP*, 2014. 3.2.2

[161] Petrov, S., Barrett, L., Thibaux, R., and Klein, D., "Learning accurate, compact, and interpretable tree annotation," in *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pp. 433–440, Association for Computational Linguistics, 2006. 3.2.1

[162] Pitler, E., Louis, A., and Nenkova, A., "Automatic Sense Prediction for Implicit Discourse Relations in Text," in *ACL-IJCNLP*, 2009. 2.2.2, 2.4.1, 3.2.3, 3.2.3.2, 5, 4.3.5, 5.2.3

[163] Pitler, E., Raghupathy, M., Mehta, H., Nenkova, A., Lee, A., and Joshi, A., "Easily Identifiable Discourse Relations," in *CoLING*, 2008. 2.2.2, 4.3.5

[164] Pradhan, S. S., Hovy, E., Marcus, M., Palmer, M., Ramshaw, L., and Weischedel, R., "Ontonotes: A unified relational semantic representation," *International Journal of Semantic Computing*, vol. 1, no. 04, pp. 405–419, 2007. 3.2.2

[165] Prasad, R., Dinesh, N., Lee, A., Miltsakaki, E., Robaldo, L., Joshi, A., and Webber, B., "The Penn Discourse Treebank 2.0," in *LREC*, 2008. 2.1.2, 3.1, 3.1.3, 3.2.2, 3.2.3, 3.2.3.1, 4.3.1, 4.3.5, 5, 5.2.3

[166] Prasad, R., Joshi, A., and Webber, B., "Realization of discourse relations by other means: alternative lexicalizations," in *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pp. 1023–1031, Association for Computational Linguistics, 2010. 3.1.1

[167] Prasad, R., Miltsakaki, E., Dinesh, N., Lee, A., Joshi, A., Robaldo, L., and Webber, B. L., "The penn discourse treebank 2.0 annotation manual." The PDTB Research Group, 2007. 5.2.3

[168] Roller, B. T. C. G. D., "Max-margin markov networks," *Advances in neural information processing systems*, vol. 16, p. 25, 2004. 2.3.1

[169] Rutherford, A. and Xue, N., "Improving the Inference of Implicit Discourse Relations via Classifying Explicit Discourse Connectives," in *NAACL-HLT*, 2015. 4.3, 4.3.5, 9, 5.1, 5.2.2, 5.2.3, 7

[170] Rutherford, A. T. and Xue, N., "Discovering implicit discourse relations through brown cluster pair representation and coreference patterns," *EACL 2014*, p. 645, 2014. (document), 2.4.1, 2.4.1, 3.2.2, 3.2.3.1, 4, 4.3.5

[171] Sagae, K., "Analysis of Discourse Structure with Syntactic Dependencies and Data-Driven Shift-Reduce Parsing," in *Proceedings of the 11th International Conference on Parsing Technologies (IWPT)*, 2009. 2.2.1, 2.2.1, 2.3.1, 2.4.1, 3.1, 3.1.1, 1

[172] Shi, L. and Mihalcea, R., "Putting pieces together: Combining FrameNet, VerbNet and WordNet for robust semantic parsing," in *Computational linguistics and intelligent text processing*, pp. 100–111, Springer, 2005. 1

[173] Smolensky, P., "Tensor product variable binding and the representation of symbolic structures in connectionist systems," *Artificial intelligence*, vol. 46, no. 1, pp. 159–216, 1990. 3.2.1, 6.1

[174] Socher, R., Chen, D., Manning, C. D., and Ng, A., "Reasoning with neural tensor networks for knowledge base completion," in *Advances in Neural Information Processing Systems*, pp. 926–934, 2013. 2.4.3, 3.2.1

[175] Socher, R., Huval, B., Manning, C. D., and Ng, A. Y., "Semantic compositionality through recursive matrix-vector spaces," in *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pp. 1201–1211, Association for Computational Linguistics, 2012. 1.3.1, 2.4.3

[176] Socher, R., Karpathy, A., Le, Q. V., Manning, C. D., and Ng, A. Y., "Grounded compositional semantics for finding and describing images with sentences," *Transactions of the Association for Computational Linguistics*, vol. 2, pp. 207–218, 2014. 2.4.3

[177] Socher, R., Lin, C. C., Manning, C., and Ng, A. Y., "Parsing natural scenes and natural language with recursive neural networks," in *Proceedings of the 28th International Conference on Machine Learning*, pp. 129–136, 2011. 2.4.3, 3.2, 3.2.1, 3.2.2, 6.1, 6.1.3

[178] Socher, R., Manning, C. D., and Ng, A. Y., "Learning continuous phrase representations and syntactic parsing with recursive neural networks," in *Proceedings of the NIPS-2010 Deep Learning and Unsupervised Feature Learning Workshop*, pp. 1–9, 2010. 2.4.3

[179] Socher, R., Perelygin, A., Wu, J. Y., Chuang, J., Manning, C. D., Ng, A. Y., and Potts, C., "Recursive deep models for semantic compositionality over a sentiment treebank," in *Proceedings of the conference on empirical methods in natural language processing (EMNLP)*, 2013. (document), 3.2, 6.1, 13, 6.1.3

[180] Somasundaran, S., Namata, G., Wiebe, J., and Getoor, L., "Supervised and unsupervised methods in employing discourse relations for improving opinion polarity classification," in *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pp. 170–179, Association for Computational Linguistics, 2009. 6.1.1

[181] Sordoni, A., Galley, M., Auli, M., Brockett, C., Ji, Y., Mitchell, M., Nie, J.-Y., Gao, J., and Dolan, B., "A Neural Network Approach to Context-Sensitive Generation of Conversational Responses," in *NAACL*, June 2015. 4

[182] Soricut, R. and Marcu, D., "Sentence Level Discourse Parsing using Syntactic and Lexical Information," in *NAACL*, 2003. 2.4.1, 6.1

[183] Sporleder, C. and Lascarides, A., "Using automatically labelled examples to classify rhetorical relations: An assessment," *Natural Language Engineering*, vol. 14, no. 03, pp. 369–416, 2008. 1.4, 1.6, 5, 5.1, 5.2.3

[184] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R., "Dropout: A simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014. 4.3.5

[185] Stede, M., *Discourse processing*. Morgan & Claypool Publishers, 2011. 1, 1

[186] Stolcke, A., Ries, K., Coccaro, N., Shriberg, E., Bates, R., Jurafsky, D., Taylor, P., Martin, R., Van Ess-Dykema, C., and Meteer, M., "Dialogue act modeling for automatic tagging and recognition of conversational speech," *Computational linguistics*, vol. 26, no. 3, pp. 339–373, 2000. 4.3.2, 4.3.5, 4.3.5, 10

[187] Sutskever, I., Vinyals, O., and Le, Q. V., "Sequence to sequence learning with neural networks," in *Advances in neural information processing systems*, pp. 3104–3112, 2014. 4.2.1, 4.2.2

[188] Taboada, M., "SFU Review Corpus," 2008. http://www.sfu.ca/mtaboada/research/SFU_Review_Corpus.html. 6.1.3

[189] Taboada, M. and Mann, W. C., "Applications of rhetorical structure theory," *Discourse studies*, vol. 8, no. 4, pp. 567–588, 2006. 1

[190] Tang, D., Qin, B., and Liu, T., "Document modeling with gated recurrent neural network for sentiment classification," in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 1422–1432, 2015. 4.1

[191] Taskar, B., Guestrin, C., and Koller, D., "Max-margin Markov networks," in *Advances in neural information processing systems*, 2004. 3.1

[192] Trivedi, R. and Eisenstein, J., "Discourse Connectors for Latent Subjectivity in Sentiment Analysis," in *Proceedings of the North American Chapter of the Association for Computational Linguistics (NAACL)*, (Atlanta, GA), pp. 808–813, 2013. 6.1.1

[193] Turian, J., Ratinov, L., and Bengio, Y., "Word Representations: A Simple and General Method for Semi-Supervised Learning," in *Proceedings of ACL*, pp. 384–394, 2010. 3.1.1, 3.2.2

[194] Turney, P. D., "Thumbs up or thumbs down?: semantic orientation applied to unsupervised classification of reviews," in *Proceedings of the 40th annual meeting on association for computational linguistics*, pp. 417–424, Association for Computational Linguistics, 2002. 1.5.1

[195] Turney, P. D., Pantel, P., and others, "From frequency to meaning: Vector space models of semantics," *Journal of artificial intelligence research*, vol. 37, no. 1, pp. 141–188, 2010. 3.2

[196] Van de Cruys, T. and Apidianaki, M., "Latent semantic word sense induction and disambiguation," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*, pp. 1476–1485, Association for Computational Linguistics, 2011. 3.1.3

[197] Van der Maaten, L. and Hinton, G., "Visualizing data using t-sne," *Journal of Machine Learning Research*, vol. 9, no. 2579-2605, p. 85, 2008. (document), 3.1.3, 18

[198] Voll, K. and Taboada, M., "Not all words are created equal: Extracting semantic orientation as a function of adjective relevance," in *AI 2007: Advances in Artificial Intelligence*, pp. 337–346, Springer, 2007. (document), 24, 6.1

[199] Wachsmuth, H., Trenkmann, M., Stein, B., and Engels, G., "Modeling Review Argumentation for Robust Sentiment Analysis," in *Proceedings of the 25th International Conference on Computational Linguistics COLING*, 2014. 6.1.1

[200] Wang, F. and Wu, Y., "Exploiting hierarchical discourse structure for review sentiment analysis," in *2013 International Conference on Asian Language Processing*, 2013. 6.1.1

[201] Wang, F., Wu, Y., and Qiu, L., "Exploiting discourse relations for sentiment analysis.," in *COLING (Posters)*, pp. 1311–1320, 2012. 6.1.1

[202] Wang, T. and Cho, K., "Larger-Context Language Modelling," *arXiv preprint arXiv:1511.03729*, 2015. 4, 4.1

[203] Webb, N., Hepple, M., and Wilks, Y., "Dialogue act classification based on intra-utterance features," in *Proceedings of the AAAI Workshop on Spoken Language Understanding*, 2005. 4.3.5

[204] Webber, B., "D-LTAG: Extending lexicalized TAG to discourse," *Cognitive Science*, vol. 28, no. 5, pp. 751–779, 2004. 2.1.2, 4.3.1

[205] Webber, B. and Joshi, A., "Discourse structure and computation: past, present and future," in *Proceedings of the ACL-2012 Special Workshop on Rediscovering 50 Years of Discoveries*, pp. 42–54, Association for Computational Linguistics, 2012. 1, 1

[206] Wilson, T., Wiebe, J., and Hoffmann, P., "Recognizing contextual polarity in phrase-level sentiment analysis," in *Proceedings of the conference on human language technology and empirical methods in natural language processing*, pp. 347–354, Association for Computational Linguistics, 2005. 2.4.1, 6.1.2

[207] Yang, B. and Cardie, C., "Context-aware learning for sentence-level sentiment analysis with posterior regularization.," in *ACL (1)*, pp. 325–335, 2014. 6.1.1

[208] Ye, Y., Fossum, V. L., and Abney, S., "Latent features in automatic tense translation between chinese and english," in *Proceedings of the Fifth SIGHAN Workshop on Chinese Language Processing*, pp. 48–55, 2006. 6.2.1

[209] Young, R. M. and Moore, J. D., "Dpocl: A principled approach to discourse planning," in *Proceedings of the Seventh International Workshop on Natural Language Generation*, pp. 13–20, Association for Computational Linguistics, 1994. 2.1

[210] Yu, C.-N. J. and Joachims, T., "Learning structural SVMs with latent variables," in *Proceedings of the 26th Annual International Conference on Machine Learning*, pp. 1169–1176, ACM, 2009. 3.1.2

[211] Zaremba, W., Sutskever, I., and Vinyals, O., "Recurrent neural network regularization," *arXiv preprint arXiv:1409.2329*, 2014. 4.3.5, 4.3.5

[212] Zhou, L., Li, B., Gao, W., Wei, Z., and Wong, K.-F., "Unsupervised discovery of discourse relations for eliminating intra-sentence polarity ambiguities," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 162–171, Association for Computational Linguistics, 2011. 6.1.1

[213] Zhou, Z.-M., Xu, Y., Niu, Z.-Y., Lan, M., Su, J., and Tan, C. L., "Predicting discourse connectives for implicit discourse relation recognition," in *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pp. 1507–1514, Association for Computational Linguistics, 2010. 2.2.2, 5, 3.2.3.2

[214] Zirn, C., Niepert, M., Stuckenschmidt, H., and Strube, M., "Fine-Grained Sentiment Analysis with Structural Features.," in *IJCNLP*, pp. 336–344, 2011. 6.1.1, 6.1.3