A Systematic Study of Neural Discourse Models for Implicit Discourse Relation

Attapol T. Rutherford
Yelp
San Francisco, CA, USA
teruth@yelp.com

Vera Demberg
Saarland University
Saarbrücken, Germany
vera@coli.uni-saarland.de

Nianwen Xue
Brandeis University
Waltham, MA, USA
xuen@brandeis.edu

Abstract

Inferring implicit discourse relations in natural language text is the most difficult subtask in discourse parsing. Many neural network models have been proposed to tackle this problem. However, the comparison for this task is not unified, so we could hardly draw clear conclusions about the effectiveness of various architectures. Here, we propose neural network models that are based on feedforward and long-short term memory architecture and systematically study the effects of varying structures. To our surprise, the best-configured feedforward architecture outperforms LSTM-based model in most cases despite thorough tuning. Further, we compare our best feedforward system with competitive convolutional and recurrent networks and find that feedforward can actually be more effective. For the first time for this task, we compile and publish outputs from previous neural and non-neural systems to establish the standard for further comparison.

1 Introduction

The discourse structure of a natural language text has been analyzed and conceptualized under various frameworks (Mann and Thompson, 1988; Lascarides and Asher, 2007; Prasad et al., 2008). The Penn Discourse TreeBank (PDTB) and the Chinese Discourse Treebank (CDTB), currently the largest corpora annotated with discourse structures in English and Chinese respectively, view the discourse structure of a text as a set of discourse relations (Prasad et al., 2008; Zhou and Xue, 2012). Each discourse relation (e.g. causal or temporal) is grounded by a discourse connective (e.g. *because* or *meanwhile*) taking two text segments as arguments (Prasad et al., 2008). Implicit discourse relations are those where discourse connectives are omitted from the text and yet the discourse relations still hold.

While classifying explicit discourse relations is relatively easy, as the discourse connective itself provides a strong cue for the discourse relation (Pitler et al., 2008), the classification of implicit discourse relations has proved to be notoriously hard and remained one of the last missing pieces in an end-to-end discourse parser (Xue et al., 2015). In the absence of explicit discourse connectives, implicit discourse relations have to be inferred from their two arguments. Previous approaches on inferring implicit discourse relations have typically relied on features extracted from their two arguments. These features include the Cartesian products of the word tokens in the two arguments as well as features manually crafted from various lexicons such as verb classes and sentiment lexicons (Pitler et al., 2009; Rutherford and Xue, 2014). These lexicons are used mainly to offset the data sparsity problem created by pairs of word tokens used directly as features.

Neural network models are an attractive alternative for this task, but it is not clear how well they will fare with a small dataset, typically found in discourse annotation projects. Many neural approaches have been proposed. However, we lack a unified standard comparison to really learn whether we make any progress at all because not all past studies agree on the same experimental settings such as label sets to use. Previous work used four binary classification (Pitler et al., 2008; Rutherford and Xue, 2014) , 4-way coarse sense classification (Rutherford and Xue, 2015) , and intermediate sense classification (Lin et al., 2009). CoNLL Shared Task introduces a unified scheme for evaluation along with a new unseen test set in English in 2015 (Xue et al., 2015) and in Chinese in 2016 (Xue et al., 2016). We want to corrobo-

Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers, pages 281–291,
Valencia, Spain, April 3-7, 2017.

rate this new evaluation scheme by running more benchmark results and providing the output under this evaluation scheme. We systematically compare the relative advantages of different neural architectures and publish the outputs from the systems for the research community to conduct further analysis.

In this work, we explore multiple neural architectures in an attempt to find the best distributed representation and neural network architecture suitable for this task in both English and Chinese. We do this by probing the different points on the spectrum of structurality from structureless bag-of-words models to sequential and tree-structured models. We use feedforward, sequential long short-term memory (LSTM), and tree-structured LSTM models to represent these three points on the spectrum. To the best of our knowledge, there is no prior study that investigates the contribution of the different architectures in neural discourse analysis.

Our main contributions and findings from this work can be summarized as follows:

- We establish that the simplest feedforward discourse model outperforms systems with surface features and perform comparably with or even outperforms recurrent and convolutional architectures. This holds across different label sets in English and in Chinese.
- We investigate the contribution of the linguistic structures in neural discourse modeling and found that high-dimensional word vectors trained on a large corpus can compensate for the lack of structures in the model, given the small amount of annotated data.
- We collect and publish the system outputs from many neural architectures on the standard experimental settings for the community to conduct more error analysis. These are made available on the author's website.

2 Model Architectures

Following previous work, we assume that the two arguments of an implicit discourse relation are given so that we can focus on predicting the senses of the implicit discourse relations. The input to our model is a pair of text segments called Arg1 and Arg2, and the label is one of the senses defined in the Penn Discourse Treebank as in the example below:

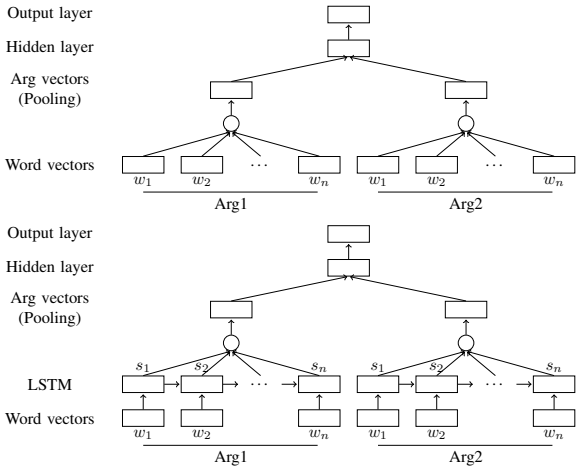

Figure 1: (Top) Feedforward architecture. (Bottom) Sequential Long Short-Term Memory architecture.

Input:

Arg1 Senator Pete Domenici calls this effort "the first gift of democracy"

Arg2 The Poles might do better to view it as a Trojan Horse.

Output:

Sense Comparison.Contrast

In all architectures, each word in the argument is represented as a k-dimensional word vector trained on an unannotated data set. We use various model architectures to transform the semantics represented by the word vectors into distributed continuous-valued features. In the rest of the section, we explain the details of the neural network architectures that we design for the implicit discourse relations classification task. The models are summarized schematically in Figure 1.

2.1 Bag-of-words Feedforward Model

This model does not model the structure or word order of a sentence. The features are simply obtained through element-wise pooling functions. Pooling is one of the key techniques in neural network modeling of computer vision (Krizhevsky et al., 2012; LeCun et al., 2010). Max pooling is known to be very effective in vision, but it is unclear what pooling function works well when it comes to pooling word vectors. Summation pooling and mean pooling have been claimed to perform well at composing meaning of a short phrase from individual word vectors (Le and Mikolov,

2014; Blacoe and Lapata, 2012; Mikolov et al., 2013b; Braud and Denis, 2015). The Arg1 vector a^1 and Arg2 vector a^2 are computed by applying element-wise pooling function f on all of the N_1 word vectors in Arg1 $w^1_{1:N_1}$ and all of the N_2 word vectors in Arg2 $w^2_{1:N_2}$ respectively:

$$a^1_i = f(w^1_{1:N_1,i})$$
$$a^2_i = f(w^2_{1:N_2,i})$$

We consider three different pooling functions namely max, summation, and mean pooling functions:

$$f_{max}(w_{1:N}, i) = \max_{j=1}^{N} w_{j,i}$$
$$f_{sum}(w_{1:N}, i) = \sum_{j=1}^{N} w_{j,i}$$
$$f_{mean}(w_{1:N}, i) = \sum_{j=1}^{N} w_{j,i}/N$$

Inter-argument interaction is modeled directly by the hidden layers that take argument vectors as features. Discourse relations cannot be determined based on the two arguments individually. Instead, the sense of the relation can only be determined when the arguments in a discourse relation are analyzed jointly. The first hidden layer h_1 is the non-linear transformation of the weighted linear combination of the argument vectors:

$$h_1 = \tanh(W_1 \cdot a^1 + W_2 \cdot a^2 + b_{h_1})$$

where W_1 and W_2 are $d \times k$ weight matrices and b_{h_1} is a d-dimensional bias vector. Further hidden layers h_t and the output layer o follow the standard feedforward neural network model.

$$h_t = \tanh(W_{h_t} \cdot h_{t-1} + b_{h_t})$$
$$o = \text{softmax}(W_o \cdot h_T + b_o)$$

where W_{h_t} is a $d \times d$ weight matrix, b_{h_t} is a d-dimensional bias vector, and T is the number of hidden layers in the network.

2.2 Sequential Long Short-Term Memory (LSTM)

A sequential Long Short-Term Memory Recurrent Neural Network (LSTM-RNN) models the semantics of a sequence of words through the use of hidden state vectors. Therefore, the word ordering does affect the resulting hidden state vectors, unlike the bag-of-word model. For each word vector at word position t, we compute the corresponding hidden state vector s_t and the memory cell vector from the previous step, using standard formula for LSTM. The argument vectors are the results of applying a pooling function over the hidden state vectors.

$$a^1_i = f(s^1_{1:N_1,i})$$
$$a^2_i = f(s^2_{1:N_2,i})$$

In addition to the three pooling functions that we describe in the previous subsection, we also consider using only the last hidden state vector, which should theoretically be able to encode the semantics of the entire word sequence.

$$f_{last}(s_{1:N,i}) = s_{N,i}$$

Inter-argument interaction and the output layer are modeled in the same fashion as the bag-of-words model once the argument vector is computed.

2.3 Tree LSTM

The principle of compositionality leads us to believe that the semantics of the argument vector should be determined by the syntactic structures and the meanings of the constituents. For a fair comparison with the sequential model, we apply the same formulation of LSTM on the binarized constituent parse tree. The hidden state vector now corresponds to a constituent in the tree. These hidden state vectors are then used in the same fashion as the sequential LSTM. The mathematical formulation is the same as Tai et al. (2015).

This model is similar to the recursive neural networks proposed by Ji and Eisenstein (2015). Our model differs from their model in several ways. We use the LSTM networks instead of the "vanilla" RNN formula and expect better results due to less complication with vanishing and exploding gradients during training. Furthermore, our purpose is to compare the influence of the model structures. Therefore, we must use LSTM cells in both sequential and tree LSTM models for a fair and meaningful comparison. The more in-depth comparison of our work and recursive neural network model by Ji and Eisenstein (2015) is provided in the discussion section.

3 Corpora and Implementation

The Penn Discourse Treebank (PDTB) We use the PDTB due to its theoretical simplicity in discourse analysis and its reasonably large size. The

Sense	Train	Dev	Test
Comparison.Concession	192	5	5
Comparison.Contrast	1612	82	127
Contingency.Cause	3376	120	197
Contingency.Pragmatic cause	56	2	5
Expansion.Alternative	153	2	15
Expansion.Conjunction	2890	115	116
Expansion.Instantiation	1132	47	69
Expansion.List	337	5	25
Expansion.Restatement	2486	101	190
Temporal.Asynchronous	543	28	12
Temporal.Synchrony	153	8	5
Total	12930	515	766

Table 1: The distribution of the level 2 sense labels in the Penn Discourse Treebank. The instances annotated with two labels are not double-counted, and partial labels are excluded.

annotation is done as another layer on the Penn Treebank on Wall Street Journal sections. Each relation consists of two spans of text that are minimally required to infer the relation, and the sense is organized hierarchically. The classification problem can be formulated in various ways based on the hierarchy. Previous work in this task has been done over three schemes of evaluation: top-level 4-way classification (Pitler et al., 2009), second-level 11-way classification (Lin et al., 2009; Ji and Eisenstein, 2015), and modified second-level classification introduced in the CoNLL 2015 Shared Task (Xue et al., 2015). We focus on the second-level 11-way classification because the labels are fine-grained enough to be useful for downstream tasks and also because the strongest neural network systems are tuned to this formulation. If an instance is annotated with two labels ($\sim$3% of the data), we only use the first label. Partial labels, which constitute $\sim$2% of the data, are excluded. Table 3 shows the distribution of labels in the training set (sections 2-21), development set (section 22), and test set (section 23).

Training Weight initialization is uniform random, following the formula recommended by Bengio (2012). The cost function is the standard cross-entropy loss function, as the hinge loss function (large-margin framework) yields consistently inferior results. We use Adagrad as the optimization algorithm of choice. The learning rates are tuned over a grid search. We monitor the accuracy on the development set to determine convergence and prevent overfitting. L2 regularization and/or dropout do not make a big impact on performance in our case, so we do not use them in the final results.

Implementation All of the models are implemented in Theano (Bergstra et al., 2010; Bastien et al., 2012). The gradient computation is done with symbolic differentiation, a functionality provided by Theano. Feedforward models and sequential LSTM models are trained on CPUs on Intel Xeon X5690 3.47GHz, using only a single core per model. A tree LSTM model is trained on a GPU on Intel Xeon CPU E5-2660. All models converge within hours.

4 Experiment on the Second-level Sense in the PDTB

We want to test the effectiveness of the inter-argument interaction and the three models described above on the fine-grained discourse relations in English. The data split and the label set are exactly the same as previous works that use this label set (Lin et al., 2009; Ji and Eisenstein, 2015).

Preprocessing All tokenization is taken from the gold standard tokenization in the PTB (Marcus et al., 1993). We use the Berkeley parser to parse all of the data (Petrov et al., 2006). We test the effects of word vector sizes. 50-dimensional and 100-dimensional word vectors are trained on the training sections of WSJ data, which is the same text as the PDTB annotation. Although this seems like too little data, 50-dimensional WSJ-trained word vectors have previously been shown to be the most effective in this task (Ji and Eisenstein, 2015). Additionally, we also test the off-the-shelf word vectors trained on billions of tokens from Google News data freely available with the `word2vec` tool. All word vectors are trained on the Skip-gram architecture (Mikolov et al., 2013b; Mikolov et al., 2013a). Other models such as GloVe and continuous bag-of-words seem to yield broadly similar results (Pennington et al., 2014). We keep the word vectors fixed, instead of fine-tuning during training.

4.1 Results

The feedforward model performs best overall among all of the neural architectures we explore (Table 2). It outperforms the recursive neural network with bilinear output layer introduced by Ji and Eisenstein (2015) ($p < 0.05$; bootstrap test) and performs comparably with the surface feature baseline (Lin et al., 2009), which uses var-

		No hidden layer				1 hidden layer				2 hidden layers			
Architecture	k	max	mean	sum	last	max	mean	sum	last	max	mean	sum	last
Feedforward	50	31.85	31.98	29.24	-	33.28	34.98	37.85	-	34.85	35.5	38.51	-
LSTM	50	31.85	32.11	34.46	31.85	34.07	33.15	36.16	34.34	36.16	35.11	37.2	35.24
Tree LSTM	50	28.59	28.32	30.93	28.72	29.89	30.15	32.5	31.59	32.11	31.2	32.5	29.63
Feedforward	100	33.29	32.77	28.72	-	36.55	35.64	37.21	-	36.55	36.29	37.47	-
LSTM	100	30.54	33.81	35.9	33.02	36.81	34.98	37.33	35.11	37.46	36.68	37.2	35.77
Tree LSTM	100	29.76	28.72	31.72	31.98	31.33	26.89	33.02	33.68	32.63	31.07	32.24	33.02
Feedforward	300	32.51	34.46	35.12	-	35.77	38.25	**39.56**	-	35.25	38.51	39.03	-
LSTM	300	28.72	34.59	35.24	34.64	38.25	36.42	37.07	35.5	**38.38**	37.72	37.2	36.29
Tree LSTM	300	28.45	31.59	32.76	26.76	33.81	32.89	33.94	32.63	32.11	32.76	**34.07**	32.50

Table 3: Compilation of all experimental configurations for 11-way classification on the PDTB test set. k is the word vector size. Bold-faced numbers indicate the best performance for each architecture, which is also shown in Table 2.

Model	Accuracy
PDTB Second-level senses	
Most frequent tag baseline	25.71
Our best tree LSTM	34.07
Ji & Eisenstein, (2015)	36.98
Our best sequential LSTM variant	38.38
Our best feedforward variant	39.56
Lin et al., (2009)	40.20

Table 2: Performance comparison across different models for second-level senses.

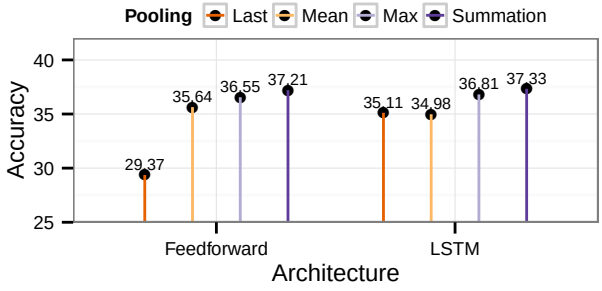

Figure 2: Summation pooling gives the best performance in general. The results are shown for the systems using 100-dimensional word vectors and one hidden layer.

ious lexical and syntactic features and extensive feature selection. Tree LSTM achieves inferior accuracy than our best feedforward model. The best configuration of the feedforward model uses 300-dimensional word vectors, one hidden layer, and the summation pooling function to derive argument feature vectors. The model behaves well during training and converges in less than an hour on a CPU.

The sequential LSTM model outperforms the feedforward model when word vectors are not high-dimensional and not trained on a large corpus (Figure 4). Moving from 50 units to 100 units trained on the same dataset, we do not observe much of a difference in performance in both architectures, but the sequential LSTM model beats the feedforward model in both settings (Table 3). This suggests that only 50 dimensions are needed for the WSJ corpus. However, the trend reverses when we move to 300-dimensional word vectors trained on a much larger corpus. These results suggest an interaction between the lexical information encoded by word vectors and the structural information encoded by the model itself.

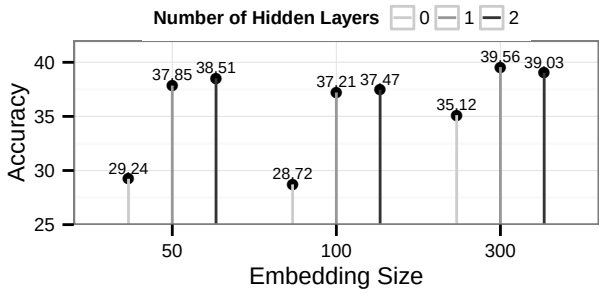

Figure 3: Inter-argument interaction can be modeled effectively with hidden layers. The results are shown for the feedforward models with summation pooling, but this effect can be observed robustly in all architectures we consider.

Hidden layers, especially the first one, make a substantial impact on performance. This effect is observed across all architectures (Figure 3). Strikingly, the improvement can be as high as 8% absolute when used with the feedforward model with small word vectors. We tried up to four hidden layers and found that the additional hidden layers yield diminishing—if not negative—returns. These effects are not an artifact of the training process as we have tuned the models quite extensively, although it might be the case that we do not

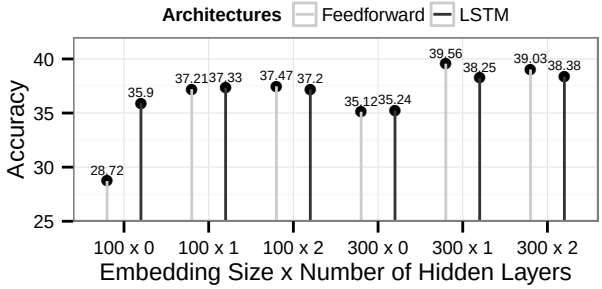

Figure 4: Comparison between feedforward and sequential LSTM when using summation pooling function.

have sufficient data to fit those extra parameters.

Summation pooling is effective for both feedforward and LSTM models (Figure 2). The word vectors we use have been claimed to have some additive properties (Mikolov et al., 2013b), so summation pooling in this experiment supports this claim. Max pooling is only effective for LSTM, probably because the values in the word vector encode the abstract features of each word relative to each other. It can be trivially shown that if all of the vectors are multiplied by -1, then the results from max pooling will be totally different, but the word similarities remain the same. The memory cells and the state vectors in the LSTM models transform the original word vectors to work well the max pooling operation, but the feedforward net cannot transform the word vectors to work well with max pooling as it is not allowed to change the word vectors themselves.

4.2 Why does the feedforward model outperform the LSTM models?

Summing up vectors indeed works better than recurrent models. We provide further evidence for this claim in Section 5. Sequential and tree LSTM models might work better if we are given larger amount of data. We observe that LSTM models outperform the feedforward model when word vectors are smaller, so it is unlikely that we train the LSTMs incorrectly. It is more likely that we do not have enough annotated data to train a more powerful model such as LSTM. In previous work, LSTMs are applied to tasks with a lot of labeled data compared to mere 12,930 instances that we have (Vinyals et al., 2015; Chiu and Nichols, 2015; İrsoy and Cardie, 2014). Another explanation comes from the fact that the contextual information encoded in the word vectors can compensate for the lack of structure in the model in this task. Word vectors are already trained to encode the words in their linguistic context especially information from word order.

Our discussion would not be complete without explaining our results in relation to the recursive neural network model proposed by Ji and Eisenstein (2015). Why do sequential LSTM models outperform recursive neural networks or tree LSTM models? Although this first comes as a surprise to us, the results are consistent with recent works that use sequential LSTM to encode syntactic information. For example, Vinyals et al. (2015) use sequential LSTM to encode the features for syntactic parse output. Tree LSTM seems to show improvement when there is a need to model long-distance dependency in the data (Tai et al., 2015; Li et al., 2015). Furthermore, the benefits of tree LSTM are not readily apparent for a model that discards the syntactic categories in the intermediate nodes and makes no distinction between heads and their dependents, which are at the core of syntactic representations.

Another point of contrast between our work and Ji and Eisenstein's (2015) is the modeling choice for inter-argument interaction. Our experimental results show that the hidden layers are an important contributor to the performance for all of our models. We choose linear inter-argument interaction instead of bilinear interaction, and this decision gives us at least two advantages. Linear interaction allows us to stack up hidden layers without the exponential growth in the number of parameters. Secondly, using linear interaction allows us to use high dimensional word vectors, which we found to be another important component for the performance. The recursive model by Ji and Eisenstein (2015) is limited to 50 units due to the bilinear layer. Our choice of linear inter-argument interaction and high-dimensional word vectors turns out to be crucial to building a competitive neural network model for classifying implicit discourse relations.

5 Extending the results across neural architectures, label sets, and languages

We want to provide further evidence that feedforward models perform well without surface features or without sophisticated recurrent or convolutional structures across different label sets and languages as well. Toward that goal, we evaluate

our models on non-explicit discourse relation data used in English and Chinese CoNLL 2016 Shared Task.

5.1 English discourse relations

We follow the experimental setting used in CoNLL 2015-2016 Shared Task. To compare our results against previous systems, we compile all of the official system outputs, and make them publicly available. The label set is modified by the shared task organizers into 15 different senses including EntRel as another sense (Xue et al., 2015; Xue et al., 2016). We use the 300-dimensional word vector used in the previous experiment and tune the number of hidden layers and hidden units on the development set. We consider the following models: Bidirectional-LSTM (Akanksha and Eisenstein, 2016), two flavors of convolutional networks (Qin et al., 2016; Wang and Lan, 2016), two variations of simple argument pooling (Mihaylov and Frank, 2016; Schenk et al., 2016), and the best system using surface features alone (Wang and Lan, 2015). The comparison results and brief system descriptions are shown in Table 4.

Our model presents the state-of-the-art system on the blind test set in English. We once again confirm that manual features are not necessary for this task and that our feedforward network outperforms the best available LSTM and convolutional networks in many settings despite its simplicity. While performing well in-domain, convolutional networks degrade sharply when tested on the blind slightly out-of-domain dataset.

5.2 Chinese discourse relations

We evaluate our model on the Chinese Discourse Treebank (CDTB) because its annotation is the most comparable to the PDTB (Zhou and Xue, 2015). The sense set consists of 10 different senses, which are not organized in a hierarchy, unlike the PDTB. We use the version of the data provided to the CoNLL 2016 Shared Task participants. This version has 16,946 instances of discourse relations total in the combined training and development sets. The test set is not yet available at the time of submission, so the system is evaluated based on the average accuracy over 7-fold cross-validation on the combined set of training and development sets.

To establish baseline comparison, we use MaxEnt models loaded with the feature sets previously shown to be effective for English, namely dependency rule pairs, production rule pairs (Lin et al., 2009), Brown cluster pairs (Rutherford and Xue, 2014), and word pairs (Marcu and Echihabi, 2002). We use information gain criteria to select the best subset of each feature set, which is crucial in feature-based discourse parsing.

Model	Acc.
CoNLL-ST 2015-2016 English (WSJ Test set)	
Most frequent tag baseline	21.36
Our best LSTM variant	31.76
Wang and Lan (2015) - winning team	34.45
Our best feedforward variant	**36.13**
CoNLL-ST 2016 Chinese (CTB Test set)	
Most frequent tag baseline	77.14
ME + Production rules	80.81
ME + Dependency rules	82.34
ME + Brown pairs (1000 clusters)	82.36
Out best LSTM variant	82.48
ME + Brown pairs (3200 clusters)	82.98
ME + Word pairs	83.13
ME + All feature sets	84.16
Our best feedforward variant	**85.45**

Table 5: Our best feedforward variant significantly outperforms the systems with surface features ($p < 0.05$). ME=Maximum Entropy model

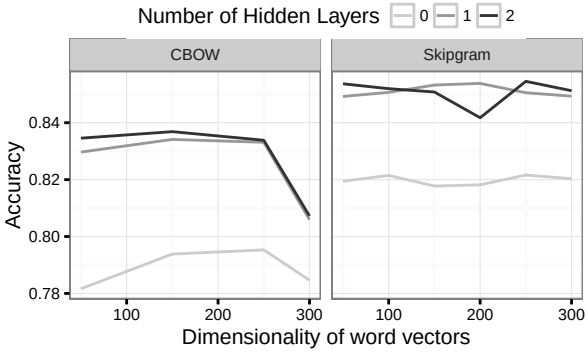

Figure 5: Comparing the accuracies across Chinese word vectors for feedforward model.

Chinese word vectors are induced through CBOW and Skipgram architecture in `word2vec` (Mikolov et al., 2013a) on Chinese Gigaword corpus (Graff and Chen, 2005) using default settings. The number of dimensions that we try are 50, 100, 150, 200, 250, and 300. We induce 1,000 and 3,000 Brown clusters on the Gigaword corpus.

Table 5 shows the results for the models which are best tuned on the number of hidden units, hidden layers, and the types of word vectors. The feedforward variant of our model significantly outperforms the strong baselines in both English and Chinese ($p < 0.05$ bootstrap test). This suggests that our approach is robust against different label

Systems	Arg vector	Features?	Blind set	WSJ Test	WSJ Dev
Ours	Summing vectors	No	**0.3767**	0.3613	0.4032
Akanksha & Eisenstein (2016)	2-layer Bi-LSTM	Yes	0.3675	0.3495	0.4072
Qin et al. (2016)	Convolutional net	No	0.3538	0.3820	0.4632
Mihaylov & Frank (2016)	Averaging vectors	Yes	0.3451	0.3919	0.4032
Schenk et al. (2016)	Avg + Product	No	0.3185	0.3761	0.4542
Wang & Lan (2016)	Convolutional net	No	0.3418	**0.4091**	**0.4642**
Wang & Lan (2015)	N/A	Yes	0.3629	0.3445	0.4272

Table 4: Comparing various systems on the CoNLL 2016 Shared Task standard datasets. Manual features are no longer needed for a competitive system. While performing well in-domain, convolutional networks degrade sharply when tested on the blind slightly out-of-domain dataset.

sets, and our findings are valid across languages. Our Chinese model outperforms all of the feature sets known to work well in English despite using only word vectors. The choice of neural architecture used for inducing Chinese word vectors turns out to be crucial. Chinese word vectors from Skip-gram model perform consistently better than the ones from CBOW model (Figure 5). These two types of word vectors do not show much difference in the English tasks.

6 Related Work

The prevailing approach for this task is to use surface features derived from various semantic lexicons (Pitler et al., 2009), reducing the number of parameters by mapping raw word tokens in the arguments of discourse relations to a limited number of entries in a semantic lexicon such as polarity and verb classes. Along the same vein, Brown cluster assignments have also been used as a general purpose lexicon that requires no human manual annotation (Rutherford and Xue, 2014). However, these solutions still suffer from the data sparsity problem and almost always require extensive feature selection to work well (Park and Cardie, 2012; Lin et al., 2009; Ji and Eisenstein, 2015). The work we report here explores the use of the expressive power of distributed representations to overcome the data sparsity problem found in the traditional feature engineering paradigm.

Neural network modeling has been explored to some extent in the context of this task. Recently, Braud and Denis (2015) tested various word vectors as features for implicit discourse relation classification and show that distributed features achieve the same level of accuracy as one-hot representations in some experimental settings. Ji et al. (2015; 2016) advance the state of the art for this task by using recursive and recurrent neural networks. In the work we report here, we systematically explore the use of different neural network architectures and show that when high-dimensional word vectors are used as input, a simple feed-forward architecture can outperform more sophisticated architectures such as sequential and tree-based LSTM networks, given the small amount of data.

Recurrent neural networks, especially LSTM networks, have changed the paradigm of deriving distributed features from a sentence (Hochreiter and Schmidhuber, 1997), but they have not been much explored in the realm of discourse parsing. LSTM models have been notably used to encode the meaning of source language sentence in neural machine translation (Cho et al., 2014; Devlin et al., 2014) and recently used to encode the meaning of an entire sentence to be used as features (Kiros et al., 2015). Many neural architectures have been explored and evaluated, but there is no single technique that is decidedly better across all tasks. The LSTM-based models such as Kiros et al. (2015) perform well across tasks but do not outperform some other strong neural baselines. Ji et al. (2016) uses a joint discourse language model to improve the performance on the coarse-grained label in the PDTB, but in our case, we would like to deduce how well LSTM fares in fine-grained implicit discourse relation classification, which is more practical for application.

7 Conclusions and future work

We report a series of experiments that systematically probe the effectiveness of various neural network architectures for the task of implicit discourse relation classification. We found that a feedforward variant of our model combined with hidden layers and high dimensional word vectors outperforms more complicated LSTM and convolutional models. We also establish that manually crafted surface features are not necessary for

this task. These results hold for different settings and different languages. In addition, we collect and compile the system outputs from all competitive systems and make it available for the research community to conduct further analysis. We encourage that researchers who work on this task to evaluate their systems under the CoNLL Shared Task 2015-2016 scheme to allow for easy comparison and progress tracking.

Acknowledgments

The first author was funded by the German Research Foundation (DFG) as part of SFB 1102: Information Density and Linguistic Encoding

References

Akanksha and Jacob Eisenstein. 2016. Shallow discourse parsing using distributed argument representations and bayesian optimization. *CoRR*, abs/1606.04503.

Frédéric Bastien, Pascal Lamblin, Razvan Pascanu, James Bergstra, Ian J. Goodfellow, Arnaud Bergeron, Nicolas Bouchard, and Yoshua Bengio. 2012. Theano: new features and speed improvements. Deep Learning and Unsupervised Feature Learning NIPS 2012 Workshop.

Yoshua Bengio. 2012. Practical recommendations for gradient-based training of deep architectures. In *Neural Networks: Tricks of the Trade*, pages 437–478. Springer.

James Bergstra, Olivier Breuleux, Frédéric Bastien, Pascal Lamblin, Razvan Pascanu, Guillaume Desjardins, Joseph Turian, David Warde-Farley, and Yoshua Bengio. 2010. Theano: a CPU and GPU math expression compiler. In *Proceedings of the Python for Scientific Computing Conference (SciPy)*, June. Oral Presentation.

William Blacoe and Mirella Lapata. 2012. 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*, pages 546–556. Association for Computational Linguistics.

Chloé Braud and Pascal Denis. 2015. Comparing word representations for implicit discourse relation classification. In *Empirical Methods in Natural Language Processing (EMNLP 2015)*.

Jason P.C. Chiu and Eric Nichols. 2015. Named entity recognition with bidirectional lstm-cnns. *arXiv preprint arXiv:1511.08308*.

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar, October. Association for Computational Linguistics.

Jacob Devlin, Rabih Zbib, Zhongqiang Huang, Thomas Lamar, Richard Schwartz, and John Makhoul. 2014. Fast and robust neural network joint models for statistical machine translation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, volume 1, pages 1370–1380.

David Graff and Ke Chen. 2005. Chinese gigaword. *LDC Catalog No.: LDC2003T09, ISBN*, 1:58563–58230.

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

Ozan İrsoy and Claire Cardie. 2014. Opinion mining with deep recurrent neural networks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 720–728.

Yangfeng Ji and Jacob Eisenstein. 2015. One vector is not enough: Entity-augmented distributed semantics for discourse relations. *Transactions of the Association for Computational Linguistics*, 3:329–344.

Yangfeng Ji, Gholamreza Haffari, and Jacob Eisenstein. 2016. A latent variable recurrent neural network for discourse relation language models. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Ryan Kiros, Yukun Zhu, Ruslan R. Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Skip-thought vectors. In *Advances in Neural Information Processing Systems*, pages 3276–3284.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105.

Alex Lascarides and Nicholas Asher. 2007. Segmented discourse representation theory: Dynamic semantics with discourse structure. In *Computing meaning*, pages 87–124. Springer.

Quoc V. Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. *arXiv preprint arXiv:1405.4053*.

Yann LeCun, Koray Kavukcuoglu, and Clément Farabet. 2010. Convolutional networks and applications in vision. In *Circuits and Systems (ISCAS), Proceedings of 2010 IEEE International Symposium on*, pages 253–256. IEEE.

Jiwei Li, Thang Luong, Dan Jurafsky, and Eduard Hovy. 2015. When are tree structures necessary for deep learning of representations? In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2304–2314, Lisbon, Portugal, September. Association for Computational Linguistics.

Ziheng Lin, Min-Yen Kan, and Hwee Tou Ng. 2009. Recognizing implicit discourse relations in the penn discourse treebank. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1-Volume 1*, pages 343–351. Association for Computational Linguistics.

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

Daniel Marcu and Abdessamad Echihabi. 2002. An unsupervised approach to recognizing discourse relations. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 368–375. Association for Computational Linguistics.

Mitchell P. Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. 1993. Building a large annotated corpus of english: The penn treebank. *Computational linguistics*, 19(2):313–330.

Todor Mihaylov and Anette Frank. 2016. Discourse relation sense classification using cross-argument semantic similarity based on word embeddings. In *Proceedings of the Twentieth Conference on Computational Natural Language Learning - Shared Task*, page 100.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeff Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, pages 3111–3119.

Joonsuk Park and Claire Cardie. 2012. 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*, pages 108–112. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. *Proceedings of the Empirical Methods in Natural Language Processing (EMNLP 2014)*, 12:1532–1543.

Slav Petrov, Leon Barrett, Romain Thibaux, and Dan Klein. 2006. 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*, pages 433–440. Association for Computational Linguistics.

Emily Pitler, Mridhula Raghupathy, Hena Mehta, Ani Nenkova, Alan Lee, and Aravind K Joshi. 2008. Easily identifiable discourse relations. *Technical Reports (CIS)*, page 884.

Emily Pitler, Annie Louis, and Ani Nenkova. 2009. Automatic sense prediction for implicit discourse relations in text. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2*, pages 683–691. Association for Computational Linguistics.

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

Lianhui Qin, Zhisong Zhang, and Hai Zhao. 2016. Shallow discourse parsing using convolutional neural network. In *Proceedings of the Twentieth Conference on Computational Natural Language Learning - Shared Task*, page 70.

Attapol T. Rutherford and Nianwen Xue. 2014. Discovering implicit discourse relations through brown cluster pair representation and coreference patterns. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2014)*, Gothenburg, Sweden, April.

Attapol Rutherford and Nianwen Xue. 2015. Improving the inference of implicit discourse relations via classifying explicit discourse connectives. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 799–808, Denver, Colorado, May–June. Association for Computational Linguistics.

Niko Schenk, Christian Chiarcos, Kathrin Donandt, Samuel Rönnqvist, Evgeny A. Stepanov, and Giuseppe Riccardi. 2016. Do we really need all those rich linguistic features? a neural network-based approach to implicit sense labeling. In *Proceedings of the Twentieth Conference on Computational Natural Language Learning - Shared Task*, page 41.

Kai Sheng Tai, Richard Socher, and Christopher D. Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1556–1566, Beijing, China, July. Association for Computational Linguistics.

Oriol Vinyals, Łukasz Kaiser, Terry Koo, Slav Petrov, Ilya Sutskever, and Geoffrey Hinton. 2015. Grammar as a foreign language. In C. Cortes, N.D. Lawrence, D.D. Lee, M. Sugiyama, R. Garnett, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 2755–2763. Curran Associates, Inc.

Jianxiang Wang and Man Lan. 2015. A refined end-to-end discourse parser. In *Proceedings of the Nineteenth Conference on Computational Natural Language Learning - Shared Task*, pages 17–24, Beijing, China, July. Association for Computational Linguistics.

Jianxiang Wang and Man Lan. 2016. Two end-to-end shallow discourse parsers for english and chinese in conll-2016 shared task. In *Proceedings of the Twentieth Conference on Computational Natural Language Learning - Shared Task*, page 33.

Nianwen Xue, Hwee Tou Ng, Sameer Pradhan, Rashmi Prasad, Christopher Bryant, and Attapol Rutherford. 2015. The conll-2015 shared task on shallow discourse parsing. In *Proceedings of the Nineteenth Conference on Computational Natural Language Learning - Shared Task*, pages 1–16, Beijing, China, July. Association for Computational Linguistics.

Nianwen Xue, Hwee Tou Ng, Sameer Pradhan, Bonnie Webber, Attapol Rutherford, Chuan Wang, and Hongmin Wang. 2016. The conll-2016 shared task on multilingual shallow discourse parsing. In *Proceedings of the Twentieth Conference on Computational Natural Language Learning - Shared Task*, Berlin, Germany, August. Association for Computational Linguistics.

Yuping Zhou and Nianwen Xue. 2012. Pdtb-style discourse annotation of chinese text. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 69–77. Association for Computational Linguistics.

Yuping Zhou and Nianwen Xue. 2015. The chinese discourse treebank: A chinese corpus annotated with discourse relations. *Language Resources and Evaluation*, 49(2):397–431.