A Model of Coherence Based on Distributed Sentence Representation

Jiwei Li[1] **and Eduard Hovy**[3]
[1]Computer Science Department, Stanford University, Stanford, CA 94305, USA
[3]Language Technology Institute, Carnegie Mellon University, Pittsburgh, PA 15213, USA
jiweil@stanford.edu ehovy@andrew.cmu.edu

Abstract

Coherence is what makes a multi-sentence text meaningful, both logically and syntactically. To solve the challenge of ordering a set of sentences into coherent order, existing approaches focus mostly on defining and using sophisticated features to capture the cross-sentence argumentation logic and syntactic relationships. But both argumentation semantics and cross-sentence syntax (such as coreference and tense rules) are very hard to formalize. In this paper, we introduce a neural network model for the coherence task based on distributed sentence representation. The proposed approach learns a syntactico-semantic representation for sentences automatically, using either recurrent or recursive neural networks. The architecture obviated the need for feature engineering, and learns sentence representations, which are to some extent able to capture the 'rules' governing coherent sentence structure. The proposed approach outperforms existing baselines and generates the state-of-art performance in standard coherence evaluation tasks[1].

1 Introduction

Coherence is a central aspect in natural language processing of multi-sentence texts. It is essential in generating readable text that the text planner compute which ordering of clauses (or sentences; we use them interchangeably in this paper) is likely to support understanding and avoid confusion. As Mann and Thompson (1988) define it,

> A text is coherent when it can be explained what role each clause plays with regard to the whole.

Several researchers in the 1980s and 1990s addressed the problem, the most influential of which include: Rhetorical Structure Theory (RST; (Mann and Thompson, 1988)), which defined about 25 relations that govern clause interdependencies and ordering and give rise to text tree structures; the stepwise assembly of semantic graphs to support adductive inference toward the best explanation (Hobbs et al., 1988); Discourse Representation Theory (DRT; (Lascarides and Asher, 1991)), a formal semantic model of discourse contexts that constrain coreference and quantification scoping; the model of intention-oriented conversation blocks and their stack-based queueing to model attention flow (Grosz and Sidner, 1986), and more recently an inventory of a hundred or so binary inter-clause relations and associated annotated corpus (Penn Discourse Treebank. Work in text planning implemented some of these models, especially operationalized RST (Hovy, 1988) and explanation relations (Moore and Paris, 1989) to govern the planning of coherent paragraphs. Other computational work defined so called schemas (McKeown, 1985), frames with fixed sequences of clause types to achieve stereotypical communicative intentions.

Little of this work survives. Modern research tries simply to order a collection of clauses or sentences without giving an account of which order(s) is/are coherent or what the overall text structure is. The research focuses on identifying and defining a set of increasingly sophisticated features by which algorithms can be trained to propose orderings. Features being explored include the clause entities, organized into a grid (Lapata and Barzilay, 2005; Barzilay and Lapata, 2008), coreference clues to ordering (Elsner and Charniak, 2008), named-entity categories (Eisner and Charniak, 2011), syntactic features (Louis and Nenkova, 2012), and others. Besides being time-intensive (feature engineering usually requites considerable

[1]Code available at `stanford.edu/~jiweil/` or by request from the first author.

Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 2039–2048,
October 25-29, 2014, Doha, Qatar.

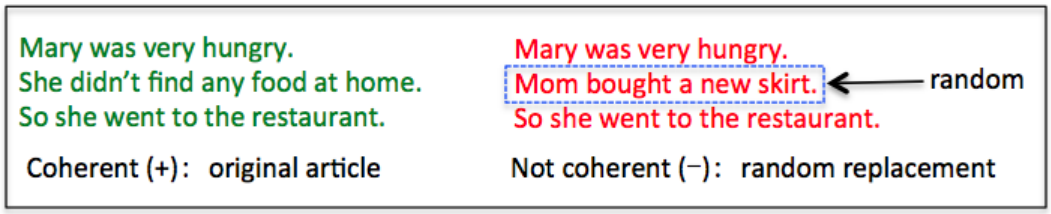

Figure 1: Illustrations of coherent (positive) vs not-coherent (negative) training examples.

effort and can depend greatly on upstream feature extraction algorithms), it is not immediately apparent which aspects of a clause or a coherent text to consider when deciding on ordering. More importantly, the features developed to date are still incapable of fully specifying the acceptable ordering(s) within a context, let alone describe why they are coherent.

Recently, deep architectures, have been applied to various natural language processing tasks (see Section 2). Such deep connectionist architectures learn a dense, low-dimensional representation of their problem in a hierarchical way that is capable of capturing both semantic and syntactic aspects of tokens (e.g., (Bengio et al., 2006)), entities, N-grams (Wang and Manning, 2012), or phrases (Socher et al., 2013). More recent researches have begun looking at higher level distributed representations that transcend the token level, such as sentence-level (Le and Mikolov, 2014) or even discourse-level (Kalchbrenner and Blunsom, 2013) aspects. Just as words combine to form meaningful sentences, can we take advantage of distributional semantic representations to explore the composition of sentences to form coherent meanings in paragraphs?

In this paper, we demonstrate that it is feasible to discover the coherent structure of a text using distributed sentence representations learned in a deep learning framework. Specifically, we consider a WINDOW approach for sentences, as shown in Figure 1, where positive examples are windows of sentences selected from original articles generated by humans, and negatives examples are generated by random replacements[2]. The semantic representations for terms and sentences are obtained through optimizing the neural network framework based on these positive vs negative examples and the proposed model produces state-of-art performance in multiple standard evaluations for coherence models (Barzilay and Lee, 2004).

The rest of this paper is organized as follows: We describe related work in Section 2, then describe how to obtain a distributed representation for sentences in Section 3, and the window composition in Section 4. Experimental results are shown in Section 5, followed by a conclusion.

2 Related Work

Coherence In addition to the early computational work discussed above, local coherence was extensively studied within the modeling framework of Centering Theory (Grosz et al., 1995; Walker et al., 1998; Strube and Hahn, 1999; Poesio et al., 2004), which provides principles to form a coherence metric (Miltsakaki and Kukich, 2000; Hasler, 2004). Centering approaches suffer from a severe dependence on manually annotated input.

A recent popular approach is the entity grid model introduced by Barzilay and Lapata (2008) , in which sentences are represented by a vector of discourse entities along with their grammatical roles (e.g., subject or object). Probabilities of transitions between adjacent sentences are derived from entity features and then concatenated to a document vector representation, which is used as input to machine learning classifiers such as SVM. Many frameworks have extended the entity approach, for example, by pre-grouping entities based on semantic relatedness (Filippova and Strube, 2007) or adding more useful types of features such as coreference (Elsner and Charniak, 2008), named entities (Eisner and Charniak, 2011), and discourse relations (Lin et al., 2011).

Other systems include the global graph model (Guinaudeau and Strube, 2013) which projects entities into a global graph. Louis and Nenkova (2012) introduced an HMM system in which the coherence between adjacent sentences is modeled by a hidden Markov framework captured by the

[2]Our approach is inspired by Collobert et al.'s idea (2011) that a word and its context form a positive training sample while a random word in that same context gives a negative training sample, when training word embeddings in the deep learning framework.

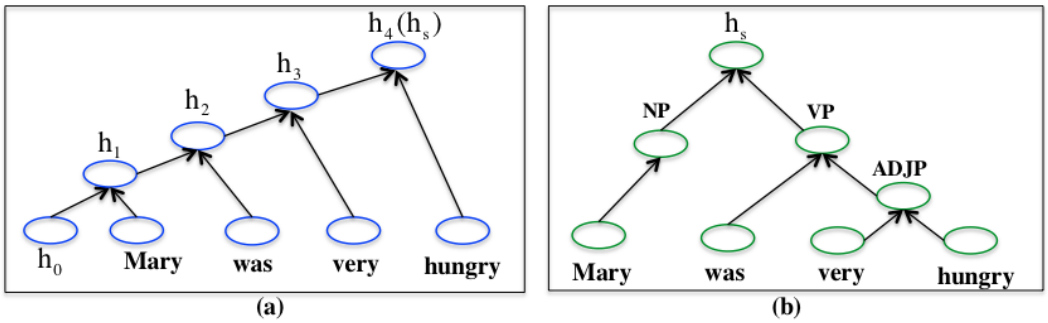

Figure 2: Sentential compositionality obtained from (a) recurrent / (b) recursive neural network. The bottom layer represents word vectors in the sentence. The top layer h_s denotes the resulting sentence vector.

transition rules of different topics.

Recurrent and Recursive Neural Networks In the context of NLP, recurrent neural networks view a sentence as a sequence of tokens and incorporate information from the past (i.e., preceding tokens) (Schuster and Paliwal, 1997; Sutskever et al., 2011) for acquisition of the current output. At each step, the recurrent network takes as input both the output of previous steps and the current token, convolutes the inputs, and forwards the result to the next step. It has been successfully applied to tasks such as language modeling (Mikolov et al., 2010) and spoken language understanding (Mesnil et al., 2013). The advantage of recurrent network is that it does not depend on external deeper structure (e.g., parse tree) and is easy to implement. However, in the recurrent framework, long-distance dependencies are difficult to capture due to the vanishing gradient problem (Bengio et al., 1994); two tokens may be structurally close to each other, even though they are far away in word sequence[3].

Recursive neural networks comprise another class of architecture, one that relies and operates on structured inputs (e.g., parse trees). It computes the representation for each parent based on its children iteratively in a bottom-up fashion. A series of variations have been proposed, each tailored to different task-specific requirements, such as Matrix-Vector RNN (Socher et al., 2012) that represents every word as both a vector and a matrix, or Recursive Neural Tensor Networks (Socher et al., 2013) that allow the model to have greater interactions between the input vectors. Many tasks have benefited from this recursive framework, including parsing (Socher et al., 2011b), sentiment analysis (Socher et al., 2013), and paraphrase detection (Socher et al., 2011a).

[3]For example, a verb and its corresponding direct object can be far away in terms of tokens if many adjectives lies in between, but they are adjacent in the parse tree (Irsoy and Cardie, 2013).

2.1 Distributed Representations

Both recurrent and recursive networks require a vector representation of each input token. Distributed representations for words were first proposed in (Rumelhart et al., 1988) and have been successful for statistical language modeling (Elman, 1990). Various deep learning architectures have been explored to learn these embeddings in an unsupervised manner from a large corpus (Bengio et al., 2006; Collobert and Weston, 2008; Mnih and Hinton, 2007; Mikolov et al., 2013), which might have different generalization capabilities and are able to capture the semantic meanings depending on the specific task at hand. These vector representations can to some extent capture interesting semantic relationships, such as $King - man \approx Queue - woman$ (Mikolov et al., 2010), and recently have been successfully used in various NLP applications, including named entity recognition, tagging, segmentation (Wang et al., 2013), and machine translation (e.g.,(Collobert and Weston, 2008; Zou et al., 2013)).

3 Sentence Model

In this section, we demonstrate the strategy adopted to compute a vector for a sentence given the sequence of its words and their embeddings. We implemented two approaches, Recurrent and Recursive neural networks, following the descriptions in for example (Mikolov et al., 2010; Sutskever et al., 2011; Socher et al., 2013). As

the details of both approaches can be readily found there, we make this section brief and omit the details for brevity.

Let s denote a sentence, comprised of a sequence of words $s = \{w_1, w_2, ..., w_{n_s}\}$, where n_s denotes the number of words within sentence s. Each word w is associated with a specific vector embedding $\mathbf{e_w} = \{e_w^1, e_w^2, ..., e_w^K\}$, where K denotes the dimension of the word embedding. We wish to compute the vector representation for current sentence $h_s = \{h_s^1, h_s^2, ..., h_s^K\}$.

Recurrent Sentence Representation (Recurrent) The recurrent network captures certain general considerations regarding sentential compositionality. As shown in Figure 2 (a), for sentence s, recurrent network successively takes word w_i at step i, combines its vector representation e_w^t with former input h_{i-1} from step $i-1$, calculates the resulting current embedding h_t, and passes it to the next step. The standard recurrent network calculates h_t as follows:

$$h_t = f(V_{Recurrent} \cdot h_{t-1} + W_{Recurrent} \cdot e_w^t + b_{Recurrent}) \quad (1)$$

where $W_{Recurrent}$ and $V_{Recurrent}$ are $K \times K$ matrixes. $b_{Recurrent}$ denotes $K \times 1$ bias vector and $f = tanh$ is a standard element-wise nonlinearity.

Note that calculation for representation at time $t = 1$ is given by:

$$h_1 = f(V_{Recurrent} \cdot h_0 + W_{Recurrent} \cdot e_w^1 + b_{Recurrent}) \quad (2)$$

where h_0 denotes the global sentence starting vector.

Recursive Sentence Representation (Recursive) Recursive sentence representation relies on the structure of parse trees, where each leaf node of the tree corresponds to a word from the original sentence. It computes a representation for each parent node based on its immediate children recursively in a bottom-up fashion until reaching the root of the tree. Concretely, for a given parent p in the tree and its two children c_1 (associated with vector representation h_{c_1}) and c_2 (associated with vector representation h_{c_2}), standard recursive networks calculates h_p for p as follows:

$$h_p = f(W_{Recursive} \cdot [h_{c_1}, h_{c_2}] + b_{Recursive}) \quad (3)$$

where $[h_{c_1}, h_{c_2}]$ denotes the concatenating vector for children vector representation h_{c_1} and h_{c_2}. $W_{Recursive}$ is a $K \times 2K$ matrix and $b_{Recursive}$ is the $1 \times K$ bias vector. $f(\cdot)$ is tanh function.

Recursive neural models compute parent vectors iteratively until the root node's representation is obtained, and use the root embedding to represent the whole sentence, as shown in Figure 2 (b).

4 Coherence Model

The proposed coherence model adopts a window approach (Collobert et al., 2011), in which we train a three-layer neural network based on a sliding windows of L sentences.

4.1 Sentence Convolution

We treat a window of sentences as a clique C and associate each clique with a tag y_C that takes the value 1 if coherent, and 0 otherwise[4]. As shown in Figure 1, cliques taken from original articles are treated as coherent and those with sentences randomly replaced are used as negative examples. .

The sentence convolution algorithm adopted in this paper is defined by a three-layer neural network, i.e., sentence-level input layer, hidden layer, and overall output layer as shown in Figure 3. Formally, each clique C takes as input a $(L \times K) \times 1$ vector h_C by concatenating the embeddings of all its contained sentences, denoted as $h_C = [h_{s_1}, h_{s_2}, ..., h_{s_L}]$. (Note that if we wish to classify the first and last sentences and include their context, we require special beginning and ending sentence vectors, which are defined as $h_{<S>}$ for s_{start} and $h_{</S>}$ for s_{end} respectively.)

Let H denote the number of neurons in the hidden (second) layer. Then each of the hidden layers takes as input h_C and performs the convolution using a non-linear tanh function, parametrized by W_{sen} and b_{sen}. The concatenating output vector for hidden layers, defined as q_C, can therefore be rewritten as:

$$q_C = f(W_{sen} \times h_C + b_{sen}) \quad (4)$$

where W_{sen} is a $H \times (L \times K)$ dimensional matrix and b_{sen} is a $H \times 1$ dimensional bias vector.

[4] instead of a binary classification (correct/incorrect), another commonly used approach is the contrastive approach that minimizes the score function $max(0, 1 - s + s_c)$ (Collobert et al., 2011; Smith and Eisner, 2005). s denotes the score of a true (coherent) window and s_c the score of a corrupt (containing incoherence) one) in an attempt to make the score of true windows larger and corrupt windows smaller. We tried the contrastive one for both recurrent and recursive networks but the binary approach constantly outperformed the contrastive one in this task.

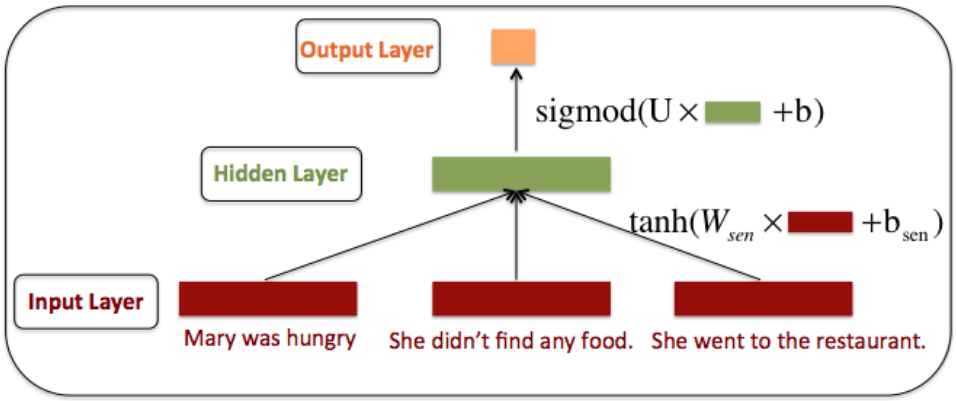

Figure 3: An example of coherence model based on a window of sentences (clique).

The output layer takes as input q_C and generates a scalar using linear function $U^T q_C + b$. A sigmod function is then adopted to project the value to a [0,1] probability space, which can be interpreted as the probability of whether one clique is coherent or not. The execution at the output layer can be summarized as:

$$p(y_C = 1) = \text{sigmod}(U^T q_C + b) \tag{5}$$

where U is an $H \times 1$ vector and b denotes the bias.

4.2 Training

In the proposed framework, suppose we have M training samples, the cost function for recurrent neural network with regularization on the training set is given by:

$$J(\Theta) = \frac{1}{M} \sum_{C \in trainset} \{-y_C \log[p(y_C = 1)] - (1 - y_C) \log[1 - p(y_C = 1)]\} + \frac{Q}{2M} \sum_{\theta \in \Theta} \theta^2 \tag{6}$$

where

$$\Theta = [W_{Recurrent}, W_{sen}, U_{sen}]$$

The regularization part is paralyzed by Q to avoid overfitting. A similar loss function is applied to the recursive network with only minor parameter altering that is excluded for brevity.

To minimize the objective $J(\Theta)$, we use the diagonal variant of AdaGrad (Duchi et al., 2011) with minibatches, which is widely applied in deep learning literature (e.g.,(Socher et al., 2011a; Pei et al., 2014)). The learning rate in AdaGrad is adapting differently for different parameters at different steps. Concretely, for parameter updates, let g_τ^i denote the subgradient at time step for parameter θ_i, which is obtained from backpropagation[5], the parameter update at time step t is given by:

$$\theta_\tau = \theta_{\tau-1} - \frac{\alpha}{\sum_{t=0}^{\tau} \sqrt{g_\tau^{i2}}} g_\tau^i \tag{7}$$

where α denotes the learning rate and is set to 0.01 in our approach. Optimal performance is achieved when batch size is set between 20 and 30.

4.3 Initialization

Elements in W_{sen} are initialized by randomly drawing from the uniform distribution $[-\epsilon, \epsilon]$, where $\epsilon = \frac{\sqrt{6}}{\sqrt{H+K \times L}}$ as suggested in (Collobert et al., 2011). $W_{recurrent}$, $V_{recurrent}$, $W_{recursive}$ and h_0 are initialized by randomly sampling from a uniform distribution $U(-0.2, 0.2)$. All bias vectors are initialized with 0. Hidden layer number H is set to 100.

Word embeddings $\{e\}$ are borrowed from Senna (Collobert et al., 2011; Collobert, 2011). The dimension for these embeddings is 50.

5 Experiments

We evaluate the proposed coherence model on two common evaluation approaches adopted in existing work (Barzilay and Lapata, 2008; Louis and Nenkova, 2012; Elsner et al., 2007; Lin et al., 2011): Sentence Ordering and Readability Assessment.

5.1 Sentence Ordering

We follow (Barzilay and Lapata, 2008; Louis and Nenkova, 2012; Elsner et al., 2007; Lin et al.,

[5]For more details on backpropagation through RNNs, see Socher et al. (2010).

2011) that all use pairs of articles, one containing the original document order and the other a random permutation of the sentences from the same document. The pairwise approach is predicated on the assumption that the original article is always more coherent than a random permutation; this assumption has been verified in Lin et al.'s work (2011).

We need to define the coherence score S_d for a given document d, where d is comprised of a series of sentences, $d = \{s_1, s_2, .., s_{N_d}\}$, and N_d denotes the number of sentences within d. Based on our clique definition, document d is comprised of N_d cliques. Taking window size $L = 3$ as example, cliques generated from document d appear as follows:

$$< s_{start}, s_1, s_2 >, < s_1, s_2, s_3 >, ..., \\ < s_{N_d-2}, s_{N_d-1}, s_{N_d} >, < s_{N_d-1}, s_{N_d}, s_{end} >$$

The coherence score for a given document S_d is the probability that all cliques within d are coherent, which is given by:

$$S_d = \prod_{C \in d} p(y_C = 1) \quad (8)$$

For document pair $< d_1, d_2 >$ in our task, we would say document d_1 is more coherent than d_2 if

$$S_{d_1} > S_{d_2} \quad (9)$$

5.1.1 Dataset

We use two corpora that are widely employed for coherence prediction (Barzilay and Lee, 2004; Barzilay and Lapata, 2008; Elsner et al., 2007). One contains reports on airplane accidents from the National Transportation Safety Board and the other contains reports about earthquakes from the Associated Press. These articles are about 10 sentences long and usually exhibit clear sentence structure. For preprocessing, we only lowercase the capital letters to match with tokens in Senna word embeddings. In the recursive network, sentences are parsed using the Stanford Parser[6] and then transformed into binary trees. The accident corpus ends up with a vocabulary size of 4758 and an average of 10.6 sentences per document. The earthquake corpus contains 3287 distinct terms and an average of 11.5 sentences per document.

For each of the two corpora, we have 100 articles for training and 100 (accidents) and 99 (earthquakes) for testing. A maximum of 20 random permutations were generated for each test article to create the pairwise data (total of 1986 test pairs for the accident corpus and 1956 for earthquakes)[7].

Positive cliques are taken from original training documents. For easy training, rather than creating negative examples by replacing centered sentences randomly, the negative dataset contains cliques where centered sentences are replaced only by other sentences within the same document.

5.1.2 Training and Testing

Despite the numerous parameters in the deep learning framework, we tune only two principal ones for each setting: window size L (tried on $\{3, 5, 7\}$) and regularization parameter Q (tried on $\{0.01, 0.1, 0.25, 0.5, 1.0, 1.25, 2.0, 2.5, 5.0\}$). We trained parameters using 10-fold cross-validation on the training data. Concretely, in each setting, 90 documents were used for training and evaluation was done on the remaining articles, following (Louis and Nenkova, 2012). After tuning, the final model was tested on the testing set.

5.1.3 Model Comparison

We report performance of recursive and recurrent networks. We also report results from some popular approaches in the literature, including:

Entity Grid Model : Grid model (Barzilay and Lapata, 2008) obtains the best performance when coreference resolution, expressive syntactic information, and salience-based features are incorporated. Entity grid models represent each sentence as a column of a grid of features and apply machine learning methods (e.g., SVM) to identify the coherent transitions based on entity features (for details of entity models see (Barzilay and Lapata, 2008)). Results are directly taken from Barzilay and Lapata's paper (2008).

HMM : Hidden-Markov approach proposed by Louis and Nenkova (2012) to model the state (cluster) transition probability in the coherent context using syntactic features. Sentences need to be clustered in advance where the number of clusters is tuned as a parameter. We directly take

[6] http://nlp.stanford.edu/software/lex-parser.shtml

[7] Permutations are downloaded from http://people.csail.mit.edu/regina/coherence/CLsubmission/.

	Acci	Earthquake	Average
Recursive	0.864	0.976	0.920
Recurrent	0.840	0.951	0.895
Entity Grid	0.904	0.872	0.888
HMM	0.822	0.938	0.880
HMM+Entity	0.842	0.911	0.877
HMM+Content	0.742	0.953	0.848
Graph	0.846	0.635	0.740

Table 1: Comparison of Different Coherence Frameworks. Reported baseline results are among the best performance regarding each approach is reprinted from prior work from (Barzilay and Lapata, 2008; Louis and Nenkova, 2012; Guinaudeau and Strube, 2013).

the results from Louis and Nenkova's paper and report the best results among different combinations of parameter and feature settings[8]. We also report performances of models from Louis and Nenkova's work that combine HMM and entity/content models in a unified framework.

Graph Based Approach : Guinaudeau and Strube (2013) extended the entity grid model to a bipartite graph representing the text, where the entity transition information needed for local coherence computation is embedded in the bipartite graph. The Graph Based Approach outperforms the original entity approach in some of feature settings (Guinaudeau and Strube, 2013).

As can be seen in Table 1, the proposed frameworks (both recurrent and recursive) obtain state-of-art performance and outperform all existing baselines by a large margin. One interpretation is that the abstract sentence vector representations computed by the deep learning framework is more powerful in capturing exactly the relevant the semantic/logical/syntactic features in coherent contexts than features or other representations developed by human feature engineering are.

Another good quality of the deep learning framework is that it can be trained easily and makes unnecessary the effort required of feature engineering. In contrast, almost all existing baselines and other coherence methods require sophisticated feature selection processes and greatly rely on external feature extraction algorithm.

The recurrent network is easier to implement than the recursive network and does not rely on external resources (i.e., parse trees), but the recursive network obtains better performance by building the convolution on parse trees rather than simply piling up terms within the sentence, which is in line with common expectation.

Both recurrent and recursive models obtain better performance on the Earthquake than the Accident dataset. Scrutiny of the corpus reveals that articles reporting earthquakes exhibit a more consistent structure: earthquake outbreak, describing the center and intensity of the earthquake, injuries and rescue operations, etc., while accident articles usually exhibit more diverse scenarios.

5.2 Readability Assessment

Barzilay and Lapata (2008) proposed a readability assessment task for stylistic judgments about the difficulty of reading a document. Their approach combines a coherence system with Schwarm and Ostendorf's (2005) readability features to classify documents into two categories, more readable (coherent) documents and less readable ones. The evaluation accesses the ability to differentiate "easy to read" documents from difficult ones of each model.

5.2.1 Dataset

Barzilay and Lapata's (2008) data corpus is from the $Encyclopedia\ Britannica$ and the $Britannica\ Elementary$, the latter being a new version targeted at children. Both versions contain 107 articles. The $Encyclopedia\ Britannica$ corpus contains an average of 83.1 sentences per document and the $Britannica\ Elementary$ contains 36.6. The encyclopedia lemmas are written by different authors and consequently vary considerably in structure and vocabulary choice. Early researchers assumed that the children version ($Britannica\ Elementary$) is easier to read, hence more coherent than documents in $Encyclopedia\ Britannica$. This is a somewhat questionable assumption that needs further investigation.

5.2.2 Training and Testing

Existing coherence approaches again apply a pairwise ranking strategy and the article associated with the higher score is considered to be the more readable. As the replacement strategy for generating negative example is apparently not well fitted to this task, we adopted the following training framework: we use all sliding windows of sentences from coherent documents (documents from $Britannica\ Elementary$) as positive examples,

[8]The details for information about parameter and feature of best setting can be found in (Louis and Nenkova, 2012).

Approach	Accuracy
Recurrent	0.803
Recursive	0.828
Graph Approach	0.786
Entity	0.509
S&O	0.786
Entity+S&O	0.888

Table 2: Comparison of Different Coherence Frameworks on Readability Assessment. Reported baselines results are are taken from (Barzilay and Lapata, 2008; Guinaudeau and Strube, 2013). S&O: Schwarm and Ostendorf (2005).

and cliques from $Encyclopedia\ Britannica$ as negative examples, and again apply Eq. 6 for training and optimization. During testing, we turn to Equations 8 and 9 for pairwise comparison. We adopted five-fold cross-validation in the same way as in (Barzilay and Lapata, 2008; Guinaudeau and Strube, 2013) for fair comparison. Parameters were tuned within each training set also using five-fold cross-validation. Parameters to tune included window size L and regularization parameter Q.

5.3 Results

We report results of the proposed approaches in the work along with entity model (Barzilay and Lapata, 2008) and graph based approach (Elsner and Charniak, 2008) in Table 2. The tabs shows that deep learning approaches again significantly outperform Entry and Global Approach baselines and are nearly comparable to the combination of entity and S&O features. Again, the recursive network outperforms the recurrent network in this task.

6 Conclusion

In this paper, we apply two neural network approaches to the sentence-ordering (coherence) task, using compositional sentence representations learned by recurrent and recursive composition. The proposed approach obtains state-of-art performance on the standard coherence evaluation tasks.

Acknowledgements

The authors want to thank Richard Socher and Pradeep Dasigi for the clarification of deep learning techniques. We also thank the three anonymous EMNLP reviewers for helpful comments.

References

Regina Barzilay and Mirella Lapata. 2008. Modeling local coherence: An entity-based approach. *Computational Linguistics*, 34(1):1–34.

Regina Barzilay and Lillian Lee. 2004. Catching the drift: Probabilistic content models, with applications to generation and summarization. In *HLT-NAACL*, pages 113–120.

Yoshua Bengio, Patrice Simard, and Paolo Frasconi. 1994. Learning long-term dependencies with gradient descent is difficult. *Neural Networks, IEEE Transactions on*, 5(2):157–166.

Yoshua Bengio, Holger Schwenk, Jean-Sébastien Senécal, Fréderic Morin, and Jean-Luc Gauvain. 2006. Neural probabilistic language models. In *Innovations in Machine Learning*, pages 137–186. Springer.

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167. ACM.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *The Journal of Machine Learning Research*, 12:2493–2537.

Ronan Collobert. 2011. Deep learning for efficient discriminative parsing. In *International Conference on Artificial Intelligence and Statistics*, number EPFL-CONF-192374.

John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *The Journal of Machine Learning Research*, 12:2121–2159.

Micha Eisner and Eugene Charniak. 2011. Extending the entity grid with entity-specific features. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers-Volume 2*, pages 125–129. Association for Computational Linguistics.

Jeffrey L Elman. 1990. Finding structure in time. *Cognitive science*, 14(2):179–211.

Micha Elsner and Eugene Charniak. 2008. Coreference-inspired coherence modeling. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies: Short Papers*, pages 41–44. Association for Computational Linguistics.

Micha Elsner, Joseph L Austerweil, and Eugene Charniak. 2007. A unified local and global model for discourse coherence. In *HLT-NAACL*, pages 436–443.

Katja Filippova and Michael Strube. 2007. Extending the entity-grid coherence model to semantically related entities. In *Proceedings of the Eleventh European Workshop on Natural Language Generation*, pages 139–142. Association for Computational Linguistics.

Barbara J Grosz and Candace L Sidner. 1986. Attention, intentions, and the structure of discourse. *Computational linguistics*, 12(3):175–204.

Barbara J Grosz, Scott Weinstein, and Aravind K Joshi. 1995. Centering: A framework for modeling the local coherence of discourse. *Computational linguistics*, 21(2):203–225.

Camille Guinaudeau and Michael Strube. 2013. Graph-based local coherence modeling. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 93–103.

Laura Hasler. 2004. An investigation into the use of centering transitions for summarisation. In *Proceedings of the 7th Annual CLUK Research Colloquium*, pages 100–107.

Jerry R Hobbs, Mark Stickel, Paul Martin, and Douglas Edwards. 1988. Interpretation as abduction. In *Proceedings of the 26th annual meeting on Association for Computational Linguistics*, pages 95–103. Association for Computational Linguistics.

Eduard H Hovy. 1988. Planning coherent multisentential text. In *Proceedings of the 26th annual meeting on Association for Computational Linguistics*, pages 163–169. Association for Computational Linguistics.

Ozan Irsoy and Claire Cardie. 2013. Bidirectional recursive neural networks for token-level labeling with structure. *arXiv preprint arXiv:1312.0493*.

Nal Kalchbrenner and Phil Blunsom. 2013. Recurrent convolutional neural networks for discourse compositionality. *arXiv preprint arXiv:1306.3584*.

Mirella Lapata and Regina Barzilay. 2005. Automatic evaluation of text coherence: Models and representations. In *IJCAI*, volume 5, pages 1085–1090.

Alex Lascarides and Nicholas Asher. 1991. Discourse relations and defeasible knowledge. In *Proceedings of the 29th annual meeting on Association for Computational Linguistics*, pages 55–62. Association for Computational Linguistics.

Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents.

Ziheng Lin, Hwee Tou Ng, and Min-Yen Kan. 2011. Automatically evaluating text coherence using discourse relations. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 997–1006. Association for Computational Linguistics.

Annie Louis and Ani Nenkova. 2012. A coherence model based on syntactic patterns. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1157–1168. 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.

Kathleen R McKeown. 1985. Discourse strategies for generating natural-language text. *Artificial Intelligence*, 27(1):1–41.

Grégoire Mesnil, Xiaodong He, Li Deng, and Yoshua Bengio. 2013. Investigation of recurrent-neural-network architectures and learning methods for spoken language understanding. Interspeech.

Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Cernockỳ, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *INTERSPEECH*, pages 1045–1048.

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

Eleni Miltsakaki and Karen Kukich. 2000. The role of centering theory's rough-shift in the teaching and evaluation of writing skills. In *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, pages 408–415. Association for Computational Linguistics.

Andriy Mnih and Geoffrey Hinton. 2007. Three new graphical models for statistical language modelling. In *Proceedings of the 24th international conference on Machine learning*, pages 641–648. ACM.

Johanna D Moore and Cecile L Paris. 1989. Planning text for advisory dialogues. In *Proceedings of the 27th annual meeting on Association for Computational Linguistics*, pages 203–211. Association for Computational Linguistics.

Wenzhe Pei, Tao Ge, and Chang Baobao. 2014. Max-margin tensor neural network for chinese word segmentation. In *Proceedings of ACL*.

Massimo Poesio, Rosemary Stevenson, Barbara Di Eugenio, and Janet Hitzeman. 2004. Centering: A parametric theory and its instantiations. *Computational linguistics*, 30(3):309–363.

David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. 1988. *Learning representations by back-propagating errors*. MIT Press, Cambridge, MA, USA.

Mike Schuster and Kuldip K Paliwal. 1997. Bidirectional recurrent neural networks. *Signal Processing, IEEE Transactions on*, 45(11):2673–2681.

Sarah E Schwarm and Mari Ostendorf. 2005. Reading level assessment using support vector machines and statistical language models. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 523–530. Association for Computational Linguistics.

Noah A Smith and Jason Eisner. 2005. Contrastive estimation: Training log-linear models on unlabeled data. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 354–362. Association for Computational Linguistics.

Richard Socher, Christopher D Manning, and Andrew Y Ng. 2010. Learning continuous phrase representations and syntactic parsing with recursive neural networks. In *Proceedings of the NIPS-2010 Deep Learning and Unsupervised Feature Learning Workshop*, pages 1–9.

Richard Socher, Eric H Huang, Jeffrey Pennington, Andrew Y Ng, and Christopher D Manning. 2011a. Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. In *NIPS*, volume 24, pages 801–809.

Richard Socher, Cliff C Lin, Chris Manning, and Andrew Y Ng. 2011b. Parsing natural scenes and natural language with recursive neural networks. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 129–136.

Richard Socher, Brody Huval, Christopher D Manning, and Andrew Y Ng. 2012. 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*, pages 1201–1211. Association for Computational Linguistics.

Richard Socher, Alex Perelygin, Jean Y Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1631–1642.

Michael Strube and Udo Hahn. 1999. Functional centering: Grounding referential coherence in information structure. *Computational linguistics*, 25(3):309–344.

Ilya Sutskever, James Martens, and Geoffrey E Hinton. 2011. Generating text with recurrent neural networks. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 1017–1024.

Marilyn A Walker, Aravind Krishna Joshi, and Ellen Friedman Prince. 1998. *Centering theory in discourse*. Oxford University Press.

Sida Wang and Christopher D Manning. 2012. Baselines and bigrams: Simple, good sentiment and topic classification. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers-Volume 2*, pages 90–94. Association for Computational Linguistics.

Houfeng Wang, Longkai Zhang, Li Li, He Zhengyan, and Ni Sun. 2013. Improving chinese word segmentation on micro-blog using rich punctuations.

Will Y Zou, Richard Socher, Daniel Cer, and Christopher D Manning. 2013. Bilingual word embeddings for phrase-based machine translation. In *Conference on Empirical Methods in Natural Language Processing (EMNLP 2013)*.