A Model of Coherence Based on Sentence Vector with Long Short-Term Memory Neural Network

Qintai Liu(ql819)
New York University
New York, NY, USA
ql819@nyu.edu

Xin Guan(xg702)
New York University
New York, NY, USA
xg702@nyu.edu

Yi Zhou(yz4525)
New York University
New York, NY, USA
yz4525@nyu.edu

Abstract

Coherence, one of the most significant metrics to evaluate text quality and readability for human writing, is also a worthy topic for researchers to investigate machine's speech understanding. It is because generating text files that could express accurately and logically not only requires researchers to understand the semantic and syntactic features, but also sets high demands on designed training methods and models.

The aim of this paper is to compare the performances of recurrent and recursive neural network models in terms of sentence representation based on previous works, and employ GRU to measure the local coherence, especially on short discourse which is composed of about 11 sentences.

1 Introduction

Beaugrande and Dressler (1996) firstly defines coherence as a 'continuity of senses' and 'the mutual access and relevance within a configuration of concepts and relations'. In the 1990s, early researchers have already constructed theoretical frameworks and experimented models to evaluate coherence from different aspects.

The entity grid model (Barzilay and Lapata, 2008) is designed to obtain the information about local coherence by collecting the distribution of entities in an article and rewarding them according to classification towards discourse entities. On top of that, studies on relationship among words become prevalent. Named entity recognition task could enhance the accuracy of extracting entity tokens (Eisner and Charniak, 2011) and syntactic features enable researchers to decompose a sentence into entities and actions (Louis and Nenkova, 2012).

Besides studies on token from a global perspective, researchers also interpret sentence composition, and explore the role of each word playing in the sentence. Rhetorical Structure Theory (RST; (Mann and Thompson, 1988)) is proposed as a hierarchical structure of texts through analysis of nucleus and satellite of text relations and then be applied in practical discourse parsing and text summarization (Marcu, 2000). Discourse Representation Theory (DRT; (Lascarides and Asher, 1991; Li and Hovy, 2014)) is to understand the textual meaning by classifying the discourse referents, which stands for subjects in the sentence, and DRS conditions, which constrains the referents actions and explains the promise of referents actions. Furthermore, improvement on co-reference resolution is advantageous to associate entities with other phrase parts, thus obviating the reference confusion brought by entity grid model (Clark and Manning, 2016).

Nevertheless, it is not sufficient to only consider dependency in individual sentences as the basis of full text coherence since apart from sentence structure, ordering and position of sentences are also unnegligible. Beyond the level of words in the sentence, there are several coherence study to date trending towards analysis from sentence-level. Paragraph vector assuages the word ordering limitation of models above by converting context to a dense vector (Le and Mikolov, 2014), while another model examines that capturing information on sentences and speaker and learning those information by recurrent neural network model could achieve an outstanding performance on discourse-level coherence (Kalchbrenner and Blunsom, 2013).

In this paper, we finally choose recursive neural network as sentence model and Gated Recurrent Unit (GRU; (Cho et al., 2014)) Neural Network as the coherence model to tackle with the coherent condition of a text in sentence representation. We first use entity grid model to detect local

coherence by analyzing noun words' transitions. To capture more details about sentence, we implement recurrent neural network (RNN) and recursive neural network (Tree-RNN) and compare their performance in terms of local coherence detection.

2 Related Works

2.1 Distributed Representations

2.1.1 Bag-Of-Words Model

The simplest way to build sentence vector is bag-of-words model where a sentence is represented as the bag of its words. Even though this model illustrates the term existence of words in documents, it puts equal weight on each word on the passage, thus neglecting some essential parts. This drawback would weaken the contextual dependency and results in ambiguous decision condition on coherence task. However, entity grid model could extract nouns, representative part in a sentence, to analyze local coherence. Therefore, we would implement entity grid model as our baseline method and check how it performs.

2.1.2 Sequence Models (Recurrent Neural Network)

Recurrent neural networks (RNN) takes a sequence of tokens within the same sentence as input (Schuster and Paliwal, 2005; Sutskever et al., 2011; Li and Hovy, 2014) and is able to deal with any sequences with arbitrary length. Therefore RNN is widely considered as one of the best candidates for sequence models. Several NLP tasks, including spoken language understanding (Mesnil et al., 2013) and coherence detection (Li and Hovy, 2014) was implemented on RNN to generate sentence vector and obtains better performance, compared with previous frameworks. However, one of limitations of RNN is the vanishing/exploding gradient. In details, if the sentence is relatively long, some information at the beginning of the sentence, such as the subject of the sentence, would be lost in convoluted calculations. In recent past, a more advanced version of recurrent neural network called Gated Recurrent Unit(GRU), is prevalent in the fields of both NLP and NLU due to its ability to keep track of long-term dependencies.

2.1.3 Tree-Structured Models

Tree-structured models build sentence representation from its subcomponents representation which are phrases. The representation of each phrase is composed of its sub-phrases. After applying this transformation recursively, the constituency-based parse tree is actually generated. One of the most famous tree-structured models is recursive neural networks, which is widely used todays NLP and NLU tasks. For example, the sentence vector generated by recursive neural network is applied to sentiment classification (Socher et al., 2013) and coherence detection (Li and Hovy, 2014). Recently, a more complicated tree-structured model plays an important role in natural language processing, which is tree-structured Long Short-Term Memory Networks (Tree-LSTM). Since Tree-LSTM combines the advantage on long-term dependencies of LSTM, and the advantage of parse tree that shows the syntactic structure of a sentence. Based on the paper (Tai et al., 2015), Tree-LSTM has a really good performance in semantic relatedness.

2.2 Coherence

2.2.1 Sentence Convolution Algorithm

Sentence convolution algorithm applies a simple neural network structure consisting of an input layer, a hidden layer and an output layer (Li and Hovy, 2014). This framework takes a sequence of sentences $S_c = [S_1, \cdots, S_L]$ as the input for the input layer. Then a tanh function is applied to the linear function $W \times S_c + b$ in the hidden layer. The execution in the hidden layer is presented as follows:

$$H_c = \tanh(W \times S_c + b) \quad (1)$$

Finally, the output layer performs a linear transformation to H_c and uses sigmoid function to project the output into the range of $[0, 1]$, which could be regarded as the probability of a set of sentences to be coherent or not. This method applies a window method to deal with the sentences, which provides us a new aspect for to construct coherent models applying recursive sentence representations.

3 Baseline Model - Entity Grid Model

Entity grid model is a common strategy which uses a vector to represent sentences in coherence tasks, in particular on local coherence detection. This method is based on the claim that coherent

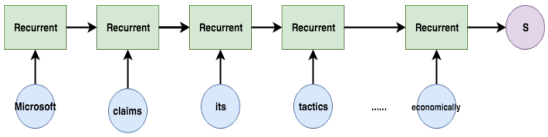

Figure 1: Recurrent neural network representation.

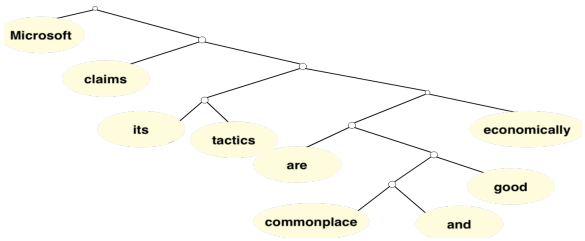

Figure 2: Recursive neural network representation.

nouns transitions are salient enough to generalize the overall coherence in the passage (Barzilay and Lapata, 2008). The entity grid model (Barzilay and Lapata, 2008) first identities discourse entity in each sentence, which is a class of coherent noun phrases, to form the vector representation for the sentence. To simplify, we use 0/1 to indicate whether the noun appear in this sentence or not and form the grid. And calculate the relative frequency of 4 possible transitions $(0-1, 0-0, 1-0, 1-1)$ of entity in adjacent sentences to generate the article vector. Finally, the article vector will be fitted into SVM to obtain the coherence score for the article.

The model is representative because it evaluates the coherence of given text by computing distributions of entity transition types between sentences. The insight we get from this entity grids method is that transitions of entity between sentences is important to coherence detection. When building sentence vector, we should consider whether the vector can underscore informative entity appropriately.

4 Sentence Model

We introduced three models to construct sentence model in Section 2.1. Since bag-of-words model is unable to memorize enough information about a sentence, we would not choose it. Our sentence model would apply recurrent neural network (RNN) and recursive neural network (Tree-RNN) respectively to compute a vector for a sentence given a sequence of words' embedding and compare their performance. Our assumption is that Tree-RNN would outperform RNN in terms of sentence representation for local coherence detection. The idea is inspired by entity grids model, which emphasizes the transition of entities between adjacent sentences. If we use Tree-RNN to build the sentence representation, generated vector could capture the syntactic and grammatical structure of the sentence. An entity, subject or object, tends to be much closer to the root than other components like adverb or complement in the parse tree, which means during back propagation, these important components of the sentence would be reviewed first. In Figure1 and Figure2, we show potentially why recursive neural networks may outperform recurrent neural networks in terms of sentence representation for coherence tasks.

Suppose we are trying to parse sentence, 'Microsoft claims its tactics are commonplace and good economically', as a vector. Figure1 shows how to do that in recurrent neural network. The restriction of this method is that essential components in the sentence may not be well incorporated in the sentence vector. Since the sentence is relatively long, the importance of Microsoft, the main subject that appears in the beginning, would decline when the sentence vector is fed to the coherence model. While the central information of the sentence could not be transferred to the end well, Figure2 representing the binary constituency tree of the sentence, has an comparative advantage on merging important parts. By observing the tree structure, we find that the level set is related with the importance of sentence configuration. More important the token is, more closely it would be to root node. Thus, the hierarchical structure could better accumulate information about key words and decrease the influence of insignificant parts at the same time.

Now we will show how to use RNN and Tree-RNN to compute a vector for a sentence.

Lets denote the embedding of a word w as $V_w = [V_{w_1}, V_{w_2}, ..., V_{w_{300}}]$. Suppose a sentence S is composed of a sequence of words $S = [W_1, W_2, ..., W_{ns}]$ with length ns.

4.1 Recurrent Sentence Representation

For sentence S, RNN consecutively takes word w_t's embedding V_{w_t} and the output h_{t-1} from previous step as input to calculate the current embedding h_t by the following formula:

$$h_t = f(W_{Recurrent} \cdot [V_{w_t}, h_{t-1}] + b_{Recurrent}) \quad (2)$$

where $W_{Recurrent}$ is a 300*600 matrix, $[V_{w_t}, h_{t-1}]$ represents the concatenation of vector V_{w_t} and h_{t-1}, so the concatenated vector has dimension 600*1, $b_{Recurrent}$ is the bias term with dimension 300*1, and the activation function f is tanh.

4.2 Recursive Sentence Representation

The input for Tree-RNN should be a parsed binary tree. Let's demonstrate how recursive neural networks works using Figure2 as an example. we compute the first parent in the bottom, denoted as p, whose left node c_1 is "commonplace" and right node c_2 is "and", by the following formula:

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

where $W_{Recursive}$ is a 300*600 matrix, $[h_{c_1}, h_{c_2}]$ represents the concatenation of vector h_{c_1} and vector h_{c_2}, so its product would be a 600*1 vector, $b_{Recursive}$ is the bias term with dimension 300*1, and the activation function f is tanh. In our example, $h_{c_1} = V_{w_{cmp}}$, $h_{c_2} = V_{w_{and}}$, h_p is the vector representation of their parents. Then we keep calculating finished nodes parent by previous formula until we obtain the root nodes vector.

After computing the recurrent or recursive representation for all of sentences in the text, we could obtain the embeddings $t = \{S_1, S_2, ..., S_{n_t}\}$ of the input text where n_t of the number of sentences in the text and S_i denotes the ith sentences vector calculated by RNN or Tree-RNN. The embeddings t constitute the text memory which is available to be accessed by latter coherence model.

5 Coherence Model

5.1 Gated Recurrent Unit (GRU)

GRU is a variation of LSTM. Instead of having a separate memory cells, GRU deals with the information flow inside the the same gating unit. Same with LSTM, GRU is also capable of dealing with long-term dependencies and solve vanishing gradient problem. Moreover, GRU is simpler and more efficient than LSTM in terms of convergence speed and generalization (Chung et al., 2014) .

The input for GRU is a sequence of sentence vectors, developed by either recurrent or recursive sentence representation we introduced before. The embeddings of a text is denoted by $t = [S_1, S_2, ..., S_{n_t}]$ and the dimension of each sentence vector is 300 and n_t is the number of sentences T has. The formulas below explain how to implement GRU neural network:

The update gate decides what information from the past would be passed to the next cell state.

$$z_t = \sigma(W_z \cdot [h_{t-1}, S_t]) \quad (4)$$

The reset gate determine what information would be discarded from previous cell states.

$$r_t = \sigma(W_r \cdot [h_{t-1}, S_t]) \quad (5)$$

Then the current memory content update the stories the latest important information by using the reset gate.

$$\tilde{h_t} = \tanh(W_h \cdot [r_t \times h_{t-1}, S_t]) \quad (6)$$

Finally, the final cell state updates the information gained from the current unit and passed it to the next cell.

$$h_t = (1 - z_t) \times h_{t-1} + z_t \times \tilde{h_t} \quad (7)$$

The output for GRU would be the vector h_{n_t} with dimension 300*1, which is the vector representation of t. The final output layer takes h_{n_t} as input and then transform it to a scalar and uses Sigmod function to project it to the probability of whether the input text is coherent or not. The following formula can summarize the final output layer:

$$p(y_T = 1) = \sinh(U^T h_{t_n} + b) \quad (8)$$

where U is a 300*1 vector

5.2 Training

Based on the previous framework, our model will take N training sample texts, each of which is composed of about 10 sentences and has target value 0 or 1. 1 represents coherence and 0 represents not coherence. Basically, we use loss function with L2 regularization to train our model. Below is the detailed objective function we use:

$$J(\theta) = \frac{1}{N} \Sigma_{T \in trainset} \{-y_T \log[p(y_T = 1)] - (1 - y_T) \log[1 - (y_T = 1)]\} + Q \Sigma_{\theta \in \Theta} \theta^2 \quad (9)$$

where

for recurrent network model

$$\Theta = [W_{Recurrent}, W_z, W_r, W_h, U]$$

for recursive network model

$$\Theta = [W_{Recursive}, W_z, W_r, W_h, U]$$

We apply Adam algorithm (Kingma and Ba, 2014) to minimize the objective function $J(\theta)$.

6 Experiment

The way we test and evaluate the coherence model proposed in this paper is to use a common evaluation approach widely employed for local coherence detection (Barzilay and Lapata, 2008; Louis and Nenkova, 2012; Elsner et al., 2007; Lin et al., 2011).

6.1 Sentence Ordering

The proposed coherence model's output for a given document d is defined as the coherence score S_d, where d is composed of a sequence of sentences, $d = \{s_1, s_2, ..., s_{N_d}\}$, where N_d denotes the number of sentences d has. In other words, the coherence sore for a given document d represents how likely d is coherent. It has been proved that original passage has a higher coherence score than a misordered one with random permutation (Lin et al., 2011). Thus, we apply the same method as researchers before (Li and Hovy, 2014; Barzilay and Lapata, 2008; Louis and Nenkova, 2012; Elsner et al., 2007; Lin et al., 2011), which is to take pairwise samples as input. Each sample is comprised of an article not been changed and an article with same contents but changed sentence order by random permutation. In terms of our task, for a document pair $< d_1, d_2 >$, we will consider d_1 is more coherent than d_2 if $S_{d_1} > S_{d_2}$

6.1.1 Dataset

The two corpora, Associated Press earthquake reports and the airplane accidents articles from the official government, are chosen because they are commonly used for coherence prediction (Barzilay and Lee, 2004; Barzilay and Lapata, 2008; Elsner et al., 2007) and show clear sentence structure. These two corpora have similar sentence length with mean value of 11.5 and 10.4 respectively and comparable vocabulary size of 3287 and 4758. During preprocessing, we simply lowercase all words and match each of them with a pre-trained word vectors, Glove (Pennington et al., 2014). In order to build binary tree, each sentence is parsed by Stanford Parser into pennTrees and then we use regular expression to remove useless information to form the final output.

To generate training and testing data, for each corpora, we take 100 articles of each corpora as training set, and have 99 articles about earthquake and 100 articles about accidents as testing set. For each article, a maximum of 20 random permutations were generated to create the pairwise data, each of which is composed of an original document and a randomly permutated one. Totally, we have 1490 training pairs for the earthquake articles and 1960 training pairs for the accident articles, and have 1720 testing pairs for earthquake and 1990 testing pairs for accident.

6.1.2 Training and Testing

For our baseline model (SVM), there are 3 number of parameters we tune which are penalty parameter C, kernel and tolerance for stopping criterion. For penalty parameter C, we try $\{0.00001, 0.0001, 0.001, 0.01, 0.1, 1, 10\}$. For kernel, we try $\{'linear', 'poly', 'rbf'\}$. For tolerance, we try $\{0.001, 0.0001\}$

For the two neural network models, there are 3 number of parameters we tune which are learning rate and drop rate and regularization parameter. For learning rate, we try $\{0.001, 0.005, 0.01\}$. The drop rate is the probability of an element of the input to be zeroed, and we try $\{0, 0.3, 0.5, 0.7\}$. For the regularization, we try $\{0, 0.001, 0.01\}$

For all of models, we use 5-fold cross-validation on training data to decide the best combination of parameters. After selecting the best parameters, each model is evaluated on the testing set.

7 Results and Analysis

Model	Accident	Earthquake	Average
Entity Gird+SVM	0.803	0.838	0.820
Recurrent NN+GRU	0.841	0.943	0.892
Recursive NN+GRU	0.898	0.997	0.948

Table 1: Comparison of Different Coherence Models

As we can see from Table1, except for our baseline model (entity grid model), the frameworks (both recurrent and recursive) we propose in the paper have pretty good performance, especially

for the framework (recursive+GRU) which obtains state-of-art performance.

7.1 Model Comparison

For the entity grid model, since it only extracts entities (nouns) within each sentences for an article and only focuses on the local entity transition, the information captured by this model is too limited to fully reflect the coherence of an article. But one advantage of this model is that the entity grid of an article is easy to be computed and training time of SVM is much less than that of neural network. In contrast of entity grid model, the sentence representation computed by either RNN or Tree-RNN do not require feature engineering and performs better in local coherence detection. The reason is that the sentence vector obtained by deep learning is able to appropriately capture both semantic and syntactic meaning of sentence.

Before we actually conduct these experiments, we suppose that the recursive based sentence vector would exceed recurrent based sentence vector in terms of local coherence detection due to the tree structure of recursive one. After the experiment, the result exactly matches our assumption. The transcendence of recursive sentence representation is due to the convolution on the binary parse tree which is able to capture the syntactic structure of a sentence. However, recurrent sentence representation is easier to implement compared with Tree-RNN because it dose not depend on the parse tree and it could also feed each word in a sentence to the model sequentially.

Additionally, the result of the experiment shows that GRU is suitable for local coherence detection especially for short articles with about 10 sentences. With a sequence of sentence vectors as the input, GRU is able to capture the relationship among adjacent sentences as well as distant sentences.

7.2 Dataset Comparison

Another intriguing result we notice in Table1 is that all of the models performance better on the Earthquake articles than the Accident articles. After checking original passages, we find that reports about earthquake usually develop in a similar structure. Since the origin of earthquakes is usually natural cause, most reports only focus on cost and extent of the damages and following rescues. However, airplane accidents may be caused by diverse reasons, including natural and human errors, or even design defects. Therefore, reports about this topic may have different emphasis and such reports are more difficult for models to capture coherent structures than earthquake reports.

8 Conclusion

In this paper, we apply entity grid, recurrent neural network and recursive neural network approaches to a local coherence detection (sentence-ordering) task. We use entity, recurrent and recursive sentence representation respectively to represent a sentence. For entity sentence representation, we calculate the relative frequency of 4 possible transitions of entity within an article to generate the vector representation of the article and then feed it to SVM to calculate the coherence score. For recurrent and recursive sentence representation, we use GRU to generate the article representation vector and convert the vector to the probability of the coherence of the article. The recursive NN+GNU achieves state-of-art performance on sentence-ordering tasks.

Our models could be applied to evaluate the quality of a text in terms of coherence, such as automatic summaries, and be productized and introduced into industry.

9 Supplemental Statements

- Everyone took part in choosing sentence model and coherence model
- Qintai and Yi applied CoreNLP to generate binary parse tree and prepare the data for Tree-RNN
- Xin developed entity grid model
- Yi developed the RNN+GRU model
- Qintai and Xin developed Tree-RNN+GRU model
- Everyone endeavored equally on final paper
- The code and datasets are included in our GitHub repository(https://github.com/qltf8/ds1012)

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.

Robert De Beaugrande and Wolfgang Dressler. 1996. *Introduction to Text Linguistics*. New York, NY.

Kyunghyun Cho, Bart van Merrie nboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder-decoder approaches. arXiv:1409.1259.

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengi. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv:1412.3555.

Kevin Clark and Christopher Manning. 2016. Improving coreference resolution by learning entity-level distributed representations. In *ACL 2016*.

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: (Volume 2: Short Papers)*, pages 125–129. Association for Computational Linguistics.

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

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

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. arXiv:1412.6980.

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.

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

Jiwei Li and Eduard Hovy. 2014. A model of coherence based on distributed sentence representation. In *Proceedings of Empirical Methods in Natural Language Processing*, volume arXiv:1510.03055.

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. 8(3):243–281.

Daniel Marcu. 2000. *The theory and practice of discourse parsing and summarization*. MIT Press, Cambridge, UK.

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

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

Mike Schuster and Kuldip K Paliwal. 2005. Bidirectional recurrent neural networks. *Signal Processing, IEEE Transactions on*, 45(11):2673–2681.

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

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.

Kai Sheng Tai, Richard Socher, and Christopher D. Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. arXiv: 1503.00075.