Pay Attention to the Ending:
Strong Neural Baselines for the ROC Story Cloze Task

Zheng Cai[1] **Lifu Tu**[2] **Kevin Gimpel**[2]

[1]University of Chicago, Chicago, IL, 60637, USA
[2]Toyota Technological Institute at Chicago, Chicago, IL, 60637, USA

jontsai@uchicago.edu, {lifu,kgimpel}@ttic.edu

Abstract

We consider the ROC story cloze task (Mostafazadeh et al., 2016) and present several findings. We develop a model that uses hierarchical recurrent networks with attention to encode the sentences in the story and score candidate endings. By discarding the large training set and only training on the validation set, we achieve an accuracy of 74.7%. Even when we discard the story plots (sentences before the ending) and only train to choose the better of two endings, we can still reach 72.5%. We then analyze this "ending-only" task setting. We estimate human accuracy to be 78% and find several types of clues that lead to this high accuracy, including those related to sentiment, negation, and general ending likelihood regardless of the story context.

1 Introduction

The ROC story cloze task (Mostafazadeh et al., 2016) tests a system's ability to choose the more plausible of two endings to a story. The incorrect ending is written to still fit the world of the story, e.g., the protagonist typically appears in both endings. The task is designed to test for "commonsense" knowledge, where the difference between the two endings lies in the plausibility of the characters' actions. The best system of Mostafazadeh et al. (2016) achieves 58.5% accuracy.

The ROC training and evaluation data differ in a key way. The training set contains 5-sentence stories. But the evaluation datasets (the validation and test sets) contain both a correct ending and an incorrect ending. This means that the task is one of outlier detection: systems must estimate the density of correct endings in the training data and then detect which of the two endings is an outlier. This becomes difficult when the evaluation contains distractors that are still somewhat plausible. For example, a model may place mass on stories that consistently mention the same characters, but this will not be useful for the task because even the incorrect ending uses the correct character names.

In this paper, we discard the 50k training stories and train only on the 1871-story validation set. We develop several neural models based on recurrent networks, comparing flat and hierarchical models for encoding the sentences in the story. We also use an attention mechanism based on the ending to identify useful parts of the plot. Our final model achieves 74.7% on the test set, outperforming all systems of Mostafazadeh et al. (2016) and approaching the state of the art results of concurrent work (Schwartz et al., 2017b).

We then discard the first four sentences of each story and use our model to score endings alone. We achieve 72.5% on the test set, outperforming most prior work without even looking at the story plots. We do a small-scale manual study of this ending-only task, finding that humans can identify the better ending in approximately 78% of cases. We report several reasons for the high accuracy of this ending-only setting, including some that are readily captured by automatic methods, such as sentiment analysis and the presence of negation words, as well as others that are more difficult, like those derived from world knowledge. Our results and analysis, combined with the similar concurrent observations of Schwartz et al. (2017a), suggest that any meaningful system for the ROC task must outperform the best ending-only baselines.

2 Task and Datasets

We refer to a 5-sentence sequence as a **story**, the incomplete 4-sentence sequence as a **plot**, and the

Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Short Papers), pages 616–622
Vancouver, Canada, July 30 - August 4, 2017. ©2017 Association for Computational Linguistics
https://doi.org/10.18653/v1/P17-2097

fifth sentence as an **ending**. The ROC story corpus (Mostafazadeh et al., 2016) contains training, validation, and test sets. The training set contains 5-sentence stories. The validation and test sets contain 4-sentence plots followed by two candidate endings, with only one correct.

Mostafazadeh et al. (2016) evaluated several methods for solving the task. Since the training set does not contain incorrect endings, their methods are based on computing similarity between the plot and ending. Their best results were obtained with the Deep Structured Semantic Model (DSSM) (Huang et al., 2013) which represents texts using character trigram counts followed by neural network layers and a similarity function.

Concurrently with our work, the LSDSem 2017 shared task was held (Mostafazadeh et al., 2017), focusing on the ROC story cloze task. Several of the participants made similar observations to what we describe here, namely that supervised learning on the validation set is more effective than learning directly from the training set, as well as noting certain biases in the endings (Schwartz et al., 2017a,b; Bugert et al., 2017; Flor and Somasundaran, 2017; Schenk and Chiarcos, 2017; Roemmele et al., 2017; Goel and Singh, 2017; Mihaylov and Frank, 2017).

3 Models and Training

We now describe our model variations. The first (EncPlotEnd) encodes the plot and ending separately, then scores them with a scoring function. The second (EncStory) concatenates the plot and ending to form a story, then encodes that story and scores its representation with a scoring function. When encoding a sequence of multiple sentences, whether with EncPlotEnd or EncStory, we consider two choices: a hierarchical encoder (Hier) that first encodes each sentence and then encodes the sentence representations, and a non-hierarchical encoder (Flat) that simply encodes the concatenation of all sentences. We also consider the possibility of including an ending-oriented attention mechanism (Att). For training, we use a simple supervised hinge loss objective.

3.1 Encoders

Our encoders encode text sequences into representations. When using our Hier model, we use a hierarchical recurrent neural network (RNN) (Li et al., 2015) with two levels. The first RNN encodes the sequence of words in a sentence; the same RNN is used for sentences in the plot and for each candidate ending. The second RNN encodes the sequence of sentence representations in a plot or story. When using our Flat model, we only use the first RNN described above; the only change is that the input becomes the concatenation of multiple sentences (separated by sentence boundary tokens).

Below we use i as a subscript to index sentences in the story or plot, and j as a superscript to index individual words in sentences. E.g., we use $\boldsymbol{w}_i$ to indicate the ith sentence of the story/plot and we use ${\boldsymbol{w}_i}^{(j)}$ to denote the word embedding vector of the jth word in the ith sentence.

3.1.1 Encoding Word Sequences

We use a bidirectional long short-term memory (BiLSTM) RNN (Hochreiter and Schmidhuber, 1997) to encode a sentence. For sentence $\boldsymbol{w}_i$:

$$\begin{aligned}\boldsymbol{f}_i &= \text{forward-LSTM}_1(\boldsymbol{w}_i)\\ \boldsymbol{b}_i &= \text{backward-LSTM}_1(\boldsymbol{w}_i)\end{aligned}$$

where $\boldsymbol{f}_i$ and $\boldsymbol{b}_i$ are hidden vector sequences. We add the forward and backward vectors at each step to obtain vectors $\boldsymbol{h}_i$, then average to obtain sentence representation $\boldsymbol{S}_i$:

$$\boldsymbol{h}_i = \boldsymbol{f}_i + \boldsymbol{b}_i \qquad \boldsymbol{S}_i = \frac{1}{|\boldsymbol{w}_i|}\sum_{j=1}^{|\boldsymbol{w}_i|} \boldsymbol{h}_i^{(j)} \tag{1}$$

We define this function from word sequence $\boldsymbol{w}_i$ to sentence representation $\boldsymbol{S}_i$ by $\text{EncWords}(\boldsymbol{w}_i)$.

3.1.2 Adding Attention

Attention mechanisms (Bahdanau et al., 2015; Mnih et al., 2014) have yielded considerable performance gains for machine comprehension (Hermann et al., 2015; Sukhbaatar et al., 2015; Chen et al., 2016), parsing (Vinyals et al., 2015), and machine translation (Luong et al., 2015).

After generating the representation $\boldsymbol{e} = \boldsymbol{S}_5 = \text{EncWords}(\boldsymbol{w}_5)$ for candidate ending $\boldsymbol{w}_5$, we use it to compute the attention over the individual hidden vectors of each sentence to compute modified sentence representations $\boldsymbol{S}_i^{\dagger}$. That is:

$$\alpha_i^{(j)} = \boldsymbol{e}^{\top}\boldsymbol{M}\boldsymbol{h}_i^{(j)} \qquad \beta_i^{(j)} \propto \exp\{\alpha_i^{(j)}\}$$

$$\boldsymbol{S}_i^{\dagger} = \sum_{j=1}^{|\boldsymbol{w}_i|} \beta_i^{(j)} \boldsymbol{h}_i^{(j)} \tag{2}$$

where $\boldsymbol{h}_i^{(j)}$ is the jth entry of $\boldsymbol{h}_i$ and $\boldsymbol{M}$ is a bilinear attention matrix.[1] Figure 1 shows this architecture. We define this attention-augmented encoder as $\textsc{AttEncWords}(\boldsymbol{w}_i, \boldsymbol{e})$.

3.1.3 Encoding Sentence Sequences

We use another BiLSTM to encode the sequence $\boldsymbol{S}$ of sentence representations $\boldsymbol{S}_i$:

$$\begin{aligned}\boldsymbol{F} &= \text{forward-LSTM}_2(\boldsymbol{S})\\ \boldsymbol{B} &= \text{backward-LSTM}_2(\boldsymbol{S})\\ \textsc{EncSents}(\boldsymbol{S}) &= \boldsymbol{F}_{\text{-}1} + \boldsymbol{B}_{\text{-}1}\end{aligned}$$

where $\boldsymbol{F}_{\text{-}1}$ is the final hidden vector in $\boldsymbol{F}$. We also use this encoder to encode the ending $\boldsymbol{e}$ by treating it as a sequence containing only one element.

3.2 Model Variations

Given our encoders, we now define the final representations $\boldsymbol{D}$ for our modeling variations, combining each of HIER and FLAT with each of ENCSTORY and ENCPLOTEND:

$$\begin{aligned}&\boldsymbol{w}_1^k = \langle \boldsymbol{w}_1, ..., \boldsymbol{w}_{k-1}, \boldsymbol{w}_k\rangle \qquad \boldsymbol{S}_1^k = \langle \boldsymbol{S}_1, ..., \boldsymbol{S}_k\rangle\\ &\boldsymbol{D}_{\textsc{FlatS}} = \textsc{EncWords}(\boldsymbol{w}_1^5)\\ &\boldsymbol{S}_i = \textsc{EncWords}(\boldsymbol{w}_i)\\ &\boldsymbol{D}_{\textsc{FlatPE}} = \langle \textsc{EncWords}(\boldsymbol{w}_1^4), \boldsymbol{S}_5\rangle\\ &\boldsymbol{D}_{\textsc{HierS}} = \textsc{EncSents}(\boldsymbol{S}_1^5)\\ &\boldsymbol{D}_{\textsc{HierPE}} = \langle \textsc{EncSents}(\boldsymbol{S}_1^4), \textsc{EncSents}(\boldsymbol{S}_5)\rangle\end{aligned}$$

When using attention, we replace ENCWORDS above with ATTENCWORDS.

After encoding the story as $\boldsymbol{D}$, we use a feed-forward network to act as a score function that takes $\boldsymbol{D}$ as input and generates a one-dimensional (scalar) output. We use tanh as the activation function on each layer of the feed-forward network and tune the numbers of hidden layers and the layer widths.

3.3 Training

Since we are training on the validation set which contains both correct and incorrect endings, we minimize the following hinge loss:

$$L = \max(0, -\text{score}(\boldsymbol{D}^+) + \text{score}(\boldsymbol{D}^-) + \delta)$$

where $\boldsymbol{D}^+$ is the representation of the correct story, $\boldsymbol{D}^-$ is the representation of the incorrect story, and $\delta = 1$ is the margin.

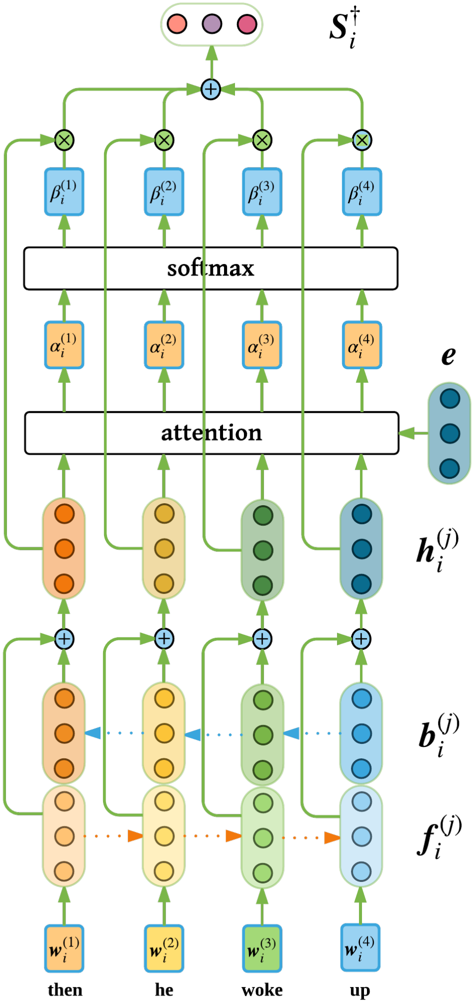

Figure 1: Attention-augmented BiLSTM for encoding a 4-word sentence $\boldsymbol{w}_i$ into a 3-dimensional representation $\boldsymbol{S}_i^\dagger$. The attention function uses the ending representation $\boldsymbol{e}$.

4 Experimental Setup

We shuffle and split the validation set into 5 folds and do 5-fold cross validation. For modeling decisions, we tune based on the average accuracy of the held-out folds. For final experiments, we choose the fold with the best held-out accuracy and report its test set accuracy. We use Adam (Kingma and Ba, 2015) for optimization with learning rate 0.001 and mini-batch size 50. We use pretrained 300-dimensional GloVe embeddings trained on Wikipedia and Gigaword (Pennington et al., 2014) and keep them fixed during training. We use L_2 regularization for the score feed-forward network, which has a single hidden layer of size 512. We use 300 for the LSTM hidden vector dimensionality for both encoders.

[1] In preliminary experiments we found bilinear attention to work better than attention based on cosine similarity.

	Flat	Hier
EncStory	79.08	80.22
EncPlotEnd	71.75	79.84

Table 1: Accuracies (%) averaged over held-out folds of 5-fold cross validation. Comparing hierarchical (Hier) and non-hierarchical (Flat) encoders, and encoding story (EncStory) vs. separately encoding plot and ending (EncPlotEnd). No attention is used.

	-Att	+Att
EncStory	80.22	79.95
EncPlotEnd	79.84	81.24

Table 2: For the EncStory and EncPlotEnd models, showing the contribution of adding attention (+Att). All models use the Hier encoder.

	val	test
DSSM‡	60.4	58.5
UW (Schwartz et al., 2017b)	-	75.2
UW (ending only)	-	72.4
trigram LM (estimated from stories)	52.4	53.6
trigram LM (estimated from endings)	53.8	54.6
Our model (Hier, EncPlotEnd, Att)	-	74.7
Our model (ending only)	-	72.5
Human‡ (story + ending)	100	100
Human (ending only)	78*	-

Table 3: Final results. * = estimate from 100; see Section 6.1. ‡ = from Mostafazadeh et al. (2016).

5 Results

Modeling Decisions. We first evaluate our modeling decisions, using the averaged held-out fold accuracy as our model selection criterion. Table 1 shows results when comparing Flat/Hier and EncStory/EncPlotEnd. Hierarchical modeling helps especially with EncPlotEnd.

Table 2 shows the contribution of attention when using Hier. Attention helps when separately encoding the plot and ending, but not when encoding the entire story. We suspect this is because when we use EncStory, the higher BiLSTM processes the sequence $\langle \boldsymbol{S}_1^\dagger, \boldsymbol{S}_2^\dagger, \boldsymbol{S}_3^\dagger, \boldsymbol{S}_4^\dagger, \boldsymbol{S}_5 \rangle$. That is, the first four sentence representations are in a different space from the ending due to the use of attention.

Final Results. Table 3 shows final results. We report the best result from Mostafazadeh et al. (2016), the best result from the concurrently-held LSDSem shared task (Schwartz et al., 2017b), and our final system configuration (with decisions tuned via cross validation as shown in Tables 1-2, then using the model with the best held-out fold accuracy). Our model achieves 74.7%, which is close to the state of the art result of 75.2%.[2]

We also report the results of stripping away the plots and running our system on just the endings ("ending only"). We use the Flat BiLSTM model on the ending followed by the feed-forward scoring function, using the same loss as above for training. We again use 5-fold cross validation on the validation set and choose the model with the highest held-out fold accuracy. We achieve 72.5%, matching the similar ending-only result of Schwartz et al. (2017b). We estimate human performance in the ending-only setting to be 78%. We provide more details in Section 6.1. These results suggest that the dataset contains systematic biases in the composition of its endings and that any meaningful system for the task must outperform the best ending-only baseline.

We also report the results of two n-gram language model baselines. We estimated trigram models using KenLM (Heafield, 2011) from two different datasets: (1) the entire training stories, and (2) only the endings from the training stories. Using only the endings works better, even though it uses one fifth of the data; this further shows the importance of focusing on endings for this task.

6 Analysis

We analyze the attention weights in our final model. Figure 2 shows the distribution of attention weights over position bins, aggregated over the plot sentences in the test set. We find that the attentions generated by the correct ending show higher weight for words early in the sentences, while the attentions for incorrect endings are higher at the ends of the sentences.

We also study the ending-only task to uncover the different kinds of bias that lead to high accuracies in this setting. We consider automatic features that can be computed on the endings and evaluate the accuracy of relying solely upon each feature as a classification rule. We then compute correlations between our ending-only model and each feature. In addition to the trigram model described above, we consider the following rules:

- **sentiment**: choose ending with higher predicted sentiment score from the Stanford sentiment an-

[2] We also tried to train the DSSM on the validation set, but were unable to approach the performance of our model. The DSSM appears to benefit greatly from the training set.

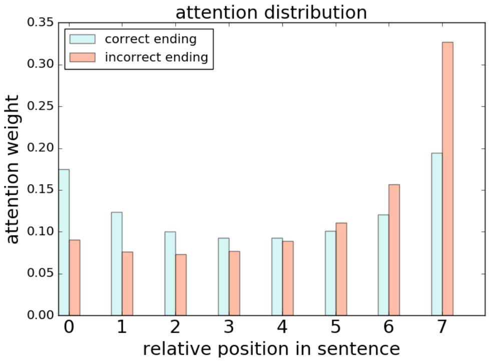

Figure 2: Attention weight distribution in test set plot sentences. Sentences were divided into eighths, then attention weights were averaged for each and normalized. For correct endings, there is more attention on early words in plot sentences.

rule	rule applicability	test accuracy	Spearman correlation
sentiment	65.9%	58.7%	0.214
negation words	20.7%	55.4%	-
length	100%	53.2%	0.047
language model	100%	54.6%	0.135

Table 4: Ending selection rules exhibiting biases in endings. Final column shows correlation between each feature and the score of our model.

alyzer (Socher et al., 2013).

- **negation**: choose ending with fewer words from {*not*, *neither*, *nor*, *never*, *n't*, *no*, *rarely*}.
- **length**: choose the longer ending.

Table 4 shows the results. Each rule yields accuracy at least 53%, with the sentiment rule nearing 59%. Even though the negation rule is only applicable in 20% of cases, its bias is strong enough to yield 5% improvement over the random baseline. These results show several reasons why an ending-only model can perform well, and suggests that our model may be identifying positive sentiment, due to its correlation of 0.214 with that feature.

We counted words in the correct and incorrect endings and in Table 5 we show some that differ between the top-50 lists for each category. E.g., "never" appears among the top 50 words in incorrect endings but not correct endings. The word count differences are accordant with the results from the sentiment and negation word rules, with non-overlapping words showing significant sentiment difference and that correct endings are more neutral or positive than incorrect ones.

correct endings:	out, !, great, new, found
incorrect endings:	n't, did, not, never, hated

Table 5: Non-overlapping words in the top 50 most frequent word list of each category.

6.1 Human Ending-Only Performance

In order to assess human performance, we randomly chose 100 ending pairs from the validation set and gave them to a human annotator, a native speaker of English who is familiar with the ROC task. The annotator was asked to select the more likely ending based only on the two endings provided. He was correct on 78, observing several kinds of cues in the endings alone in addition to those mentioned above.

In some cases, one ending sentence is simply much more likely than the other based on world knowledge. For example, the ending "the glasses fixed his headaches immediately" is much more likely than "the optometrist gave him comfortable sneakers". It is possible that the plot could change the preferred ending to the second, but this appears to be rare in the ROC dataset. In another example, "I practice all the time now" is more likely than "I hope I drop the batons" because it seems unlikely that anyone would ever hope to drop batons in the surmised world of the story. While these instances still test for a kind of "commonsense" or "world" knowledge, they do not require the plot to answer.

7 Conclusions and Future Work

Our models use none of the ROC training data but achieve strong performance, even when discarding the story plots. We uncovered several sources of bias in the endings that make the ending-only task solvable with greater than 70% accuracy. Our results suggest that any meaningful system for the ROC story cloze task should perform better than the best ending-only system. In future work, we will experiment with additional modeling choices, including adding attention to the higher BiLSTM and adding a decoder and a multi-task objective during training to improve stability.

Acknowledgments

We acknowledge Zhongtian Dai for his assistance and expertise and we thank Yejin Choi, Nasrin Mostafazadeh, Michael Roth, Roy Schwartz, and the anonymous reviewers for valuable discussions and insights.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proceedings of International Conference on Learning Representations*.

Michael Bugert, Yevgeniy Puzikov, Andreas Rücklé, Judith Eckle-Kohler, Teresa Martin, Eugenio Martínez-Cámara, Daniil Sorokin, Maxime Peyrard, and Iryna Gurevych. 2017. LSDSem 2017: Exploring data generation methods for the story cloze test. In *Proceedings of the 2nd Workshop on Linking Models of Lexical, Sentential and Discourse-level Semantics*.

Danqi Chen, Jason Bolton, and Christopher D. Manning. 2016. A thorough examination of the CNN/Daily Mail reading comprehension task. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Michael Flor and Swapna Somasundaran. 2017. Sentiment analysis and lexical cohesion for the story cloze task. In *Proceedings of the 2nd Workshop on Linking Models of Lexical, Sentential and Discourse-level Semantics*.

Pranav Goel and Anil Kumar Singh. 2017. IIT (BHU): System description for LSDSem'17 shared task. In *Proceedings of the 2nd Workshop on Linking Models of Lexical, Sentential and Discourse-level Semantics*.

Kenneth Heafield. 2011. KenLM: faster and smaller language model queries. In *Proceedings of the EMNLP 2011 Sixth Workshop on Statistical Machine Translation*.

Karl Moritz Hermann, Tomáš Kočiský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems (NIPS)*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation* 9(8).

Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. 2013. Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of the 22nd ACM International Conference on Information & Knowledge Management (CIKM)*.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of International Conference on Learning Representations*.

Jiwei Li, Minh-Thang Luong, and Dan Jurafsky. 2015. A hierarchical neural autoencoder for paragraphs and documents. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics (ACL) and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Todor Mihaylov and Anette Frank. 2017. Story cloze ending selection baselines and data examination. In *Proceedings of the 2nd Workshop on Linking Models of Lexical, Sentential and Discourse-level Semantics*.

Volodymyr Mnih, Nicolas Heess, Alex Graves, and Koray Kavukcuoglu. 2014. Recurrent models of visual attention. In *Advances in Neural Information Processing Systems (NIPS)*.

Nasrin Mostafazadeh, Nathanael Chambers, Xiaodong He, Devi Parikh, Dhruv Batra, Lucy Vanderwende, Pushmeet Kohli, and James Allen. 2016. A corpus and cloze evaluation for deeper understanding of commonsense stories. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Nasrin Mostafazadeh, Michael Roth, Annie Louis, Nathanael Chambers, and James Allen. 2017. LSDSem 2017 shared task: The story cloze test. In *Proceedings of the 2nd Workshop on Linking Models of Lexical, Sentential and Discourse-level Semantics*.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of Empirical Methods in Natural Language Processing (EMNLP)*.

Melissa Roemmele, Sosuke Kobayashi, Naoya Inoue, and Andrew Gordon. 2017. An RNN-based binary classifier for the story cloze test. In *Proceedings of the 2nd Workshop on Linking Models of Lexical, Sentential and Discourse-level Semantics*.

Niko Schenk and Christian Chiarcos. 2017. Resource-lean modeling of coherence in commonsense stories. In *Proceedings of the 2nd Workshop on Linking Models of Lexical, Sentential and Discourse-level Semantics*.

Roy Schwartz, Maarten Sap, Ioannis Konstas, Leila Zilles, Yejin Choi, and Noah A. Smith. 2017a. The effect of different writing tasks on linguistic style: A case study of the ROC story cloze task. *CoRR* abs/1702.01841.

Roy Schwartz, Maarten Sap, Ioannis Konstas, Leila Zilles, Yejin Choi, and Noah A. Smith. 2017b. Story cloze task: UW NLP system. In *Proceedings of the 2nd Workshop on Linking Models of Lexical, Sentential and Discourse-level Semantics*.

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

Sainbayar Sukhbaatar, arthur szlam, Jason Weston, and Rob Fergus. 2015. End-to-end memory networks. In *Advances in Neural Information Processing Systems (NIPS)*.

Oriol Vinyals, Ł ukasz Kaiser, Terry Koo, Slav Petrov, Ilya Sutskever, and Geoffrey Hinton. 2015. Grammar as a foreign language. In *Advances in Neural Information Processing Systems (NIPS)*.