LSDSem 2017: Exploring Data Generation Methods for the Story Cloze Test

Michael Bugert*, Yevgeniy Puzikov*, Andreas Rücklé*, Judith Eckle-Kohler*, Teresa Martin†, Eugenio Martínez-Cámara*, Daniil Sorokin*, Maxime Peyrard† and Iryna Gurevych*

*Ubiquitous Knowledge Processing Lab (UKP-TUDA)
Department of Computer Science, Technische Universität Darmstadt
www.ukp.tu-darmstadt.de

†Research Training Group AIPHES
Department of Computer Science, Technische Universität Darmstadt
www.aiphes.tu-darmstadt.de

Abstract

The Story Cloze test (Mostafazadeh et al., 2016) is a recent effort in providing a common test scenario for text understanding systems. As part of the LSDSem 2017 shared task, we present a system based on a deep learning architecture combined with a rich set of manually-crafted linguistic features. The system outperforms all known baselines for the task, suggesting that the chosen approach is promising. We additionally present two methods for generating further training data based on stories from the ROCStories corpus. Our system and generated data are publicly available on GitHub[1].

1 Introduction

The goal of the Story Cloze test is to provide a common ground for the evaluation of systems on language understanding (Mostafazadeh et al., 2016). Given four sentences of a story on everyday life events, a system has to identify the correct ending from a set of two predefined ending sentences. The "correct" ending in this case is the one which most humans would choose as the closing sentence for the story context.

We first report the discoveries we made while exploring the provided datasets (Section 2) followed by a description of our system (Section 3). We present and discuss its results (Sections 4 and 5) and come to a close in the conclusion (Section 6).

2 Dataset Exploration

Mostafazadeh et al. (2016) provide a validation and a test set for the Story Cloze test, both of which contain around 1800 stories[2]. Each of those stories consists of four context sentences and two endings to choose from. Additionally, two *ROCStories* datasets were made available with close to 100 000 stories in total. Stories from these datasets also cover everyday life events but consist of a fixed number of five sentences without ending candidates.

To gain an overview of the task, we categorized two hundred stories from the validation set based on how their correct ending can be identified.

We noticed that a large set of stories can indeed be solved via text understanding and logical inference. This includes stories where the correct ending is more likely according to script knowledge[3] (Schank and Abelson, 1977), where the topic of the wrong ending doesn't match the story context[4] or where the wrong ending contradicts the story context[5].

For some stories, the correct ending cannot be identified rationally, but rather according to commonly accepted moral values[6] or based on the reader's expectation of a positive mood in a story[7]. Regarding sentiments, we generally noticed a bias towards stories with "happy endings", i. e. stories where the sentiment expressed in the correct ending is more positive than for the wrong ending.

We infer from these observations that an approach focusing exclusively on text understanding

[1] github.com/UKPLab/lsdsem2017-story-cloze

[2] As of Feb. 2017, see cs.rochester.edu/nlp/rocstories

[3] See story 52dbbfda-5b42-4ace-8d59-55cee3eb30c0 in the Story Cloze validation set.

[4] See f8ff777f-de4d-4e3a-91bd-b197ed13f78e ibid.

[5] See a11cf506-7d19-4ab9-b0ac-a0fd85a9bd38 ibid.

[6] See 195a43c7-d43e-48e4-845b-fd6c75609df2 ibid.

[7] See 80b6447f-4c37-4194-9862-3785e5075463 ibid.

Proceedings of the 2nd Workshop on Linking Models of Lexical, Sentential and Discourse-level Semantics, pages 56–61,
Valencia, Spain, April 3, 2017.

should perform well. However, the dataset suggests that an approach which (additionally) exploits how humans write and perceive stories could also lead to respectable results, albeit not in the way originally intended for the Story Cloze test.

3 System Description

We interpret the Story Cloze test as a fully supervised learning problem, meaning we train and test our approach on instances consisting of four context sentences and two ending candidates (i. e. the format of the Story Cloze datasets). This stands in contrast to the systems reported by Mostafazadeh et al. (2016) which were trained on five-sentence stories (the ROCStories dataset).

Our approach builds around recent advances of deep learning methods and conventional feature engineering techniques.

The core of the system is a Bidirectional Recurrent Neural Network (BiRNN) (Schuster and Paliwal, 1997) with Long Short-Term Memory (LSTM) units (Hochreiter and Schmidhuber, 1997), which computes a feature representation of all given sentences. This representation is enriched by feature vectors computed over both ending candidates. To predict the correct ending of a story, we execute our neural network twice to obtain a score for each ending. These scores are compared to make a final decision. The feature vectors ensure that information on the respective other ending is available to the network during both executions.

Note that we could have instead learned ending embeddings jointly by feeding the network with both endings at the same time. In order to train a network to distinguish between correct and wrong endings, we would have had to feed the same pair another time, but with the endings swapped and the label set to its binary counterpart. However, we were concerned that such a system would eventually learn the position of the correct ending instead of its meaning. Hence, we decided against such a joint learning approach.

The following sections cover the features and neural network architecture in greater detail.

3.1 Features

We defined a set of linguistic features in order to profit from the discoveries of our dataset exploration (see Section 2). The features are:

NGramOverlap: The number of overlapping n-grams between an ending and a story context for $n = \{1, 2, 3\}$. We filtered out bigrams and trigrams using a linguistically motivated list of stopwords.

EndingLength: The token count of an ending.

Negation: The presence of words indicating negation (*none*, *never*, etc.) in an ending.

PronounContrast: The presence of pronoun mismatches between a story context and an ending. This helps to detect cases where the wrong ending is written in first person although the story context is written in third person.

SentiContrast: Disagreement in the sentiment value between the third and fourth sentences of a context and an ending. The sentiment value was computed based on a sentiment word list manually extracted from the ROCStories dataset.

For each of the features listed above, an additional feature was added to model the feature value difference between the two endings of a story. All features were extracted using the DKPro TC Framework (Daxenberger et al., 2014).

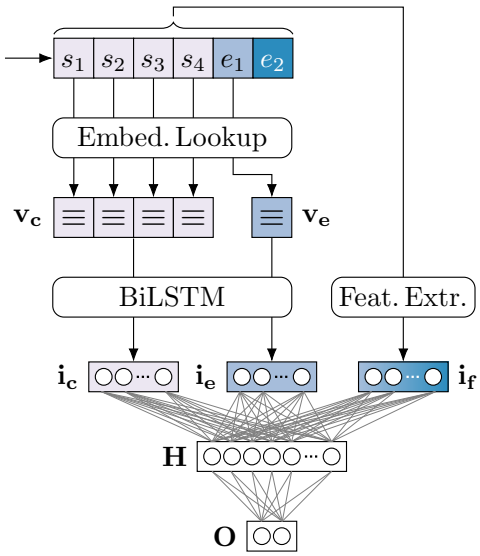

Figure 1: Neural network architecture. Depicted is the execution for obtaining a score for ending e_1.

3.2 Neural Network

The overall architecture of our neural network model is shown in Figure 1.

First, the tokens of the context sentences $s_1, \ldots, s_4$ and of ending e_1 are looked up in the word embedding matrix to obtain vectorized representations. The embeddings of the context sentences are concatenated into the vector $\mathbf{v_c}$, whereas

those of the ending form vector $\mathbf{v_e}$. $\mathbf{v_c}$ and $\mathbf{v_e}$ are fed separately to a pair of RNN networks (forward and backward) with LSTM units. This BiLSTM module encodes the meaning of context and ending in two separate vectors, $\mathbf{i_c}$ and $\mathbf{i_e}$. Given the vector of feature values $\mathbf{i_f}$, extracted externally on the four context sentences and both endings, we concatenate $\mathbf{i_c}$, $\mathbf{i_e}$ and $\mathbf{i_f}$ and feed them to the dense hidden layer $\mathbf{H}$. The output of $\mathbf{H}$ is fed to the output layer, $\mathbf{O}$. Afterwards, the softmax function is applied on the output of $\mathbf{O}$ to obtain a score representing the probability of ending e_1 being correct.

The same procedure is applied for ending e_2 in place of ending e_1. Then, the highest-scoring ending is chosen as the correct ending of the story.

Due to time constraints and system complexity we decided against hyper-parameter optimization and chose parameter values we deemed reasonable: We used pretrained 100-dimensional GloVe embeddings (Pennington et al., 2014)[8]. Following previous work on using BiLSTMs in NLP tasks (Tan et al., 2015; Tan et al., 2016; dos Santos et al., 2016), we chose a dimensionality of 141 for the LSTM output vectors and hidden layer $\mathbf{H}$.

We employ zero padding for sentences longer than 20 tokens. The network was trained in batches (size 40) for 30 epochs with the Adam optimizer (Kingma and Ba, 2014), starting with a learning rate of 10^{-4}. We apply dropout ($p = 0.3$) after computing the BiLSTM output. Finally, ReLu (Glorot et al., 2011) is used as an activation function in layer $\mathbf{H}$. The system was implemented using the TensorFlow library (Abadi et al., 2016).

3.3 Intermediate Results

We performed an intermediate evaluation of the previously described system on the Story Cloze validation set. To this end, we partitioned the dataset into a training and development split ($85\,\%$ and $15\,\%$, respectively). We compared the following systems:

BiLSTM-V: Our proposed system, trained on the $85\,\%$ training split without making use of the feature set described in Section 3.1.

BiLSTM-VF: Same as BiLSTM-V, but including the feature set.

Happy: Motivated by our dataset exploration, this baseline always picks the happier ending (the ending with the more positive sentiment). We employed the state-of-the-art sentiment annotator by Socher et al. (2013) for this purpose. This system does not take the story context into account.

The happy ending baseline reached an accuracy of 0.616 on the development split which is substantially higher than random guessing. BiLSTM-V scored 0.708 whereas BiLSTM-VF reached a slightly higher accuracy of 0.712.

3.4 Dataset Augmentation

Because our presented approach conducts supervised learning, it requires training data in the same form as testing data. Up to this point, the only dataset available in this form is the Story Cloze validation set which consists of comparably few instances for training. We experimented with ways to automatically create larger training datasets.

3.4.1 Related Work in Data Augmentation

Methods for automatic training data augmentation using unlabeled data are investigated in semi-supervised learning and have been successfully applied in many NLP tasks, for example in word sense disambiguation (Pham et al., 2005), semantic role labeling (Fürstenau and Lapata, 2012; Woodsend and Lapata, 2014) and textual entailment (Zanzotto and Pennacchiotti, 2010). Another example, which is similar to the Story Cloze test, is the task of selecting the correct answer to a question from a pool of answer candidates. To train models for this task, Feng et al. (2015) have extended a corpus of question-answer pairs by automatically adding negative answers to each pair.

We draw our inspiration from the aforementioned work and propose two methods for augmenting the ROCStories dataset in order to use it for supervised learning.

3.4.2 Shuffling

Given a ROCStory, shuffle the four context sentences, then randomly swap the fifth sentence with one of the first four. The resulting story is perceived as wrong by humans since the shuffling breaks causal and temporal relationships between the sentences. This method resembles the data quality evaluation conducted by Mostafazadeh et al. (2016).

3.4.3 KDE Sampling

In order to obtain a dataset of the same structure as the Story Cloze datasets, we heuristically complement each ROCStory with a wrong ending taken

[8] nlp.stanford.edu/data/glove.6B.zip

Story ID	Story Context	Correct Ending	Gen. Wrong Ending
3447901e-6f57 -4810-9d3c -92d72b6c0b42	A swan swam gracefully through the water. It was beautiful and white. It had a long orange beak. The crowd gathered and observed it.	It was a beautiful creature.	A large blue whale breached the water before Kathy's eyes.
0dd3e2c3-3ea4 -450a-97c4 -42a54c426018	Jane had recently gotten a new job. She was nervous about her first day of work. On the first day of work, Jane overslept. Jane arrived at work an hour late.	Jane did not make a good impression at her new job.	She told her husband that she was going all out this year.

Table 1: ROCStories with wrong endings as generated by the KDE sampling technique

from a pool of sentences. The pool can be chosen arbitrarily but for the sake of simplicity, we decided to use the existing set of ROCStory endings themselves.

Given four ROCStory context sentences, we measure the similarity between this context and each sentence from the pool (excluding the original ending). To ensure that endings match the topic and narrative of their context, our similarity measure is based on word embeddings of content words and a bag-of-words vector of occurring pronouns.

Instead of directly choosing whichever ending sentence scores the highest similarity, we attempt to replicate the characteristics of the Story Cloze datasets as close as possible. Therefore, we observe the distribution of similarity values present between each story context and its wrong ending in the validation set. Using kernel density estimation, we obtain the probability density function (PDF) of these similarity values.

To choose an ending for a ROCStory context, we then sample a similarity value from this PDF and identify the sentence in the pool whose similarity is closest to the sampled value. In a final step, we replace proper nouns occurring in the ending sentence with random proper nouns from the story context, based on POS-tags.

Table 1 shows two exemplary story endings generated using this method.

3.4.4 Comparison

We constructed two training datasets, one for each of the two augmentation methods. We then trained our BiLSTM approach (without features) for each dataset and compared their performance on the full Story Cloze validation set. The BiLSTM using the shuffled ROCStories reached an accuracy of 0.615, compared to the one using the KDE sampling method with 0.630.

From looking at the accuracies, the difference in quality between the two methods is slim. However, the quality of the shuffled stories is inferior to those of the KDE sampling method from a human point of view. For this reason, we conduct all subsequent data augmentation experiments using the KDE sampling method.

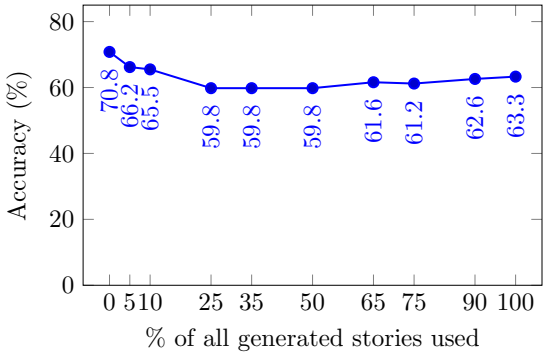

Figure 2: Accuracies on the validation set development split for BiLSTM systems trained on the training split plus a varying amount of all 98 166 generated stories.

4 Results

Following our motivation for data augmentation, we trained several BiLSTM systems on the training split of the Story Cloze validation set, augmented by a varying amount of stories generated by KDE sampling. Figure 2 shows the accuracy of these systems evaluated on the 15 % development split of the validation set. As can be seen, in all our experiments with augmentation there occurs a decrease in performance compared to the results of BiLSTM-V/BiLSTM-VF on the same data (see Section 3.3). We discuss possible reasons for this result in Section 5.

For the final evaluation, we compare the following approaches:

DSSM: The best performing baseline reported by Mostafazadeh et al. (2016).

Happy, BiLSTM-V, BiLSTM-VF: The systems previously explained in Section 3.3.

BiLSTM-T: To assess the quality of the KDE sampling method in isolation, we also include a BiLSTM system in the evaluation which is trained only on the ROCStories corpus with

Approach	Validation		Test
	Dev	Full	
DSSM	–	0.604	0.585
Happy	0.616	0.590	0.602
BiLSTM-V	0.708	–	0.701*
BiLSTM-VF	0.712	–	**0.717***
BiLSTM-T	0.637	–	0.560
BiLSTM-TF	0.634	–	0.584

Table 2: System accuracies on the Story Cloze datasets. Results marked by * are statistically significant according to McNemar's test with a *p-value* ≤ 0.05.

generated wrong endings.

BiLSTM-TF: Same as BiLSTM-T, but including the feature set.

Table 2 shows the performance of the systems on the Story Cloze test set and (if applicable) on the full validation set or its development split. It can be seen that our happy ending baseline performs comparably to the more elaborate DSSM baseline. BiLSTM-V/VF outperform the two baselines significantly. BiLSTM-T/TF fall short in accuracy compared to the ones trained on the validation set alone. The addition of features leads to improvements except for BiLSTM-T/TF when evaluated on the development split of the validation set.

5 Discussion

We manually examined several stories which were misclassified by BiLSTM-VF to identify its weaknesses. In the majority of cases, misclassified stories were sparse in sentiment or relied heavily on logical inference for identifying the correct ending. This appears plausible to us, since neither the BiLSTM nor the feature set were explicitly designed to perform advanced logical inference.

The proposed features increase the performance of our BiLSTM system. The data properties they capture are different from the distributional similarities of word embeddings and thus serve as an additional supervising signal for the neural network.

Given Figure 2 and the disparity between the results of BiLSTM-T/TF and BiLSTM-V/VF, we have to conclude that the training data augmentation did not work as well as we expected.

While the KDE sampling method creates stories resembling the ones found in the Story Cloze validation and test datasets, it does not reproduce the characteristics of these original stories well enough to produce truly valuable training data. As an example, the method does not take sentiment into account, although we demonstrated that the Story Cloze datasets are biased towards happy endings. Incorporating further characteristics into the method would amount to redefining the utilized similarity measure (no modifications would be necessary for the core idea of sampling a probability distribution).

6 Conclusion

We showed that using a sentiment-based baseline, it is trivial to reach $60\,\%$ accuracy on the Story Cloze test set without relying on text understanding. However, more sophisticated techniques are required for reaching better results. Using a deep learning architecture enriched with a set of linguistically motivated features, we surpass all previously published baselines on the task and reach 71.7% accuracy on the test set.

We proposed two methods which generate additional training data for conducting supervised learning on the Story Cloze test. To our surprise, the systems trained on generated training data performed worse than those trained on conventional training data. This could be due to our data generation not reproducing the characteristics of the original Story Cloze datasets well enough. Nonetheless, we consider the presented methods to be a valuable contribution related to the task.

Acknowledgments

We thank the anonymous reviewers for their valuable comments and insights.

This work has been supported by the German Research Foundation as part of the Research Training Group "Adaptive Preparation of Information from Heterogeneous Sources" (AIPHES) under grant No. GRK 1994/1, as part of the German-Israeli Project Cooperation (DIP, grant No. DA 1600/1-1 and grant No. GU 798/17-1), and as part of the QA-EduInf project (grant No. GU 798/18-1 and grant No. RI 803/12-1).

References

Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, Manjunath Kudlur, Josh Levenberg, Rajat Monga,

Sherry Moore, Derek G. Murray, Benoit Steiner, Paul Tucker, Vijay Vasudevan, Pete Warden, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2016. TensorFlow: A System for Large-Scale Machine Learning. In *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*, pages 265–283, GA. USENIX Association.

Johannes Daxenberger, Oliver Ferschke, Iryna Gurevych, and Torsten Zesch. 2014. DKPro TC: A Java-based Framework for Supervised Learning Experiments on Textual Data. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 61–66. Association for Computational Linguistics.

Cícero Nogueira dos Santos, Ming Tan, Bing Xiang, and Bowen Zhou. 2016. Attentive Pooling Networks. *CoRR*, abs/1602.03609.

Minwei Feng, Bing Xiang, Michael R. Glass, Lidan Wang, and Bowen Zhou. 2015. Applying deep learning to answer selection: A study and an open task. In *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, pages 813–820.

Hagen Fürstenau and Mirella Lapata. 2012. Semi-Supervised Semantic Role Labeling via Structural Alignment. *Computational Linguistics*, 38:135–171.

Xavier Glorot, Antoine Bordes, and Yoshua Bengio. 2011. Deep sparse rectifier neural networks. In *AISTATS*, volume 15, page 275.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation*, 9:1735–1780.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *CoRR*, abs/1412.6980.

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*, pages 839–849, San Diego, California. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Thanh Phong Pham, Hwee Tou Ng, and Wee Sun Lee. 2005. Word Sense Disambiguation with Semi-supervised Learning. In *Proceedings of the 20th National Conference on Artificial Intelligence - Volume 3*, AAAI'05, pages 1093–1098, Pittsburgh, Pennsylvania. AAAI Press.

Roger C. Schank and Robert P. Abelson. 1977. *Scripts, Plans, Goals, and Understanding: An Inquiry Into Human Knowledge Structures*. The Artificial Intelligence Series. Lawrence Erlbaum Associates.

Mike Schuster and Kuldip K. Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE Trans. Signal Processing*, 45:2673–2681.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Ming Tan, Bing Xiang, and Bowen Zhou. 2015. LSTM-based Deep Learning Models for non-factoid answer selection. *CoRR*, abs/1511.04108.

Ming Tan, Cicero dos Santos, Bing Xiang, and Bowen Zhou. 2016. Improved Representation Learning for Question Answer Matching. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 464–473, Berlin, Germany. Association for Computational Linguistics.

Kristian Woodsend and Mirella Lapata. 2014. Text Rewriting Improves Semantic Role Labeling. *Journal of Artificial Intelligence Research*, 51:133–164.

Fabio Massimo Zanzotto and Marco Pennacchiotti. 2010. Expanding textual entailment corpora from Wikipedia using co-training. In *Proceedings of the 2nd Workshop on The People's Web Meets NLP: Collaboratively Constructed Semantic Resources*, pages 28–36, Beijing, China. Coling 2010 Organizing Committee.