Applying BERT to Story Cloze Task

Metod Jazbec **Ada Langenfeld** **Emanuele Palumbo** **Rok Šikonja**

1 Introduction

The Story Cloze Task was first suggested in 2016 as a way to test the quality of story understanding models (Mostafazadeh et al., 2016). Given initial four sentences, the aim is to choose the correct ending among the two possible, in order to logically complete the story.

2 Methodology

As seen in (Mostafazadeh et al., 2017), approaches relying mostly on linguistic feature engineering or solely on discriminatively trained models are significantly outperformed by approaches which combine generative pre-training with discriminative fine-tuning at the end. The generative pre-training in OpenAI GPT (Radford et al., 2018) is done on a language model objective using a large unlabeled text corpus and a *Transformer* architecture (Vaswani et al., 2017) is used for a neural network model. Recently however, another approach to generative pre-training, called BERT (Devlin et al., 2018), has achieved state-of-the-art performance on a variety of natural language understanding tasks. In BERT a new objective for pre-training is proposed: a combination of a "masked language model" and a "next sentence prediction" part. This allows them to train a bidirectional Transformer, instead of a left-to-right one as done in OpenAI GPT.

Additionally, it has been shown that a further increase in accuracy can be achieved by incorporating other types of information, such as sentiment and common sense knowledge (Chen et al., 2018). As the stories are not objective text (e.g. news text), usually emotions develop throughout the story. Therefore getting a good grasp of the sentiment polarity of the story body (first four sentences of the story) can be helpful in predicting the correct ending. Incorporating commonsense knowledge is useful for discovering associations between keywords in the story body and keywords in the ending.

Based on this we decided to use a pre-trained BERT model to generate hidden representations of stories. On top of that we train a linear classifier in the form of single fully connected layer to select the correct story endings. We also combine BERT features with sentiment features and common sense distance vectors.

3 Model

3.1 BERT

3.1.1 Input representation

Since each story in the validation and test set has two candidate endings, e_1 and e_2, we generate two text sentence pairs per story: (S, e_1) and (S, e_2), where $S = (s1, s2, s3, s4)$ represents a concatenation of the first four sentences. We then use BERT's input representation for text pairs by allocating segment embedding A to tokens belonging to S and segment embedding B to the respective candidate ending's tokens. During tokenization BERT also attaches the special tokens `[CLS]`, to the beginning, and `[SEP]`, to the end, of each input sequence.

```
[CLS] Harry secretly hated haunted houses, but couldn't let his friend down.  Faking a brave
face, the went forth into the darkened house.  He managed to make it all way through without
screaming out loud.  At the very end, they fell through a trick floor into a bunch of goo.
[SEP] Harry wished he wouldn't have went.  [SEP]
```

In order to obtain a pooled representation for each input pair (S, e_i), the final hidden state (i.e. the output of the Transformer) for the [CLS] token is used. Denote this vector as $h \in \mathbb{R}^H$, where $H = 768$ is the dimension of the hidden state.

3.1.2 Absolute classification

We use pre-trained BERT weights[1] for the initialization of the Transformer. These weights are then fine-tuned during the classification task. For each (S, e_i) we thus get representation h, which we then pass through a linear layer:

$$P_B\big(y|S, e_i\big) = \mathrm{softmax}(W_B h + b_B),$$

where $y \in \{0, 1\}$ and $W_B \in \mathbb{R}^{2\times H}$. Note that this is a slightly different setting than in the Story Cloze Task, since we are trying to predict correctness for single given ending, as opposed to selecting the right ending from two options. To obtain the final prediction for each S in the test set, we then take $\arg\max_{i\in\{1,2\}}\ P_B\big(y = 1|S, e_i\big)$.

3.1.3 Relative classification

In the relative classification setting we follow an approach more related to the Story Cloze Task. First, similar to above, using BERT model we get the pooled representation for both (S, e_1) and (S, e_2), h_1 and h_2, respectively.

Then we propagate hidden representations through a linear layer with parameters $W_B \in \mathbb{R}^H$ and $b_B \in \mathbb{R}$: $z_i = W_B^\intercal h_i + b_B$.

Then we concatenate both linear outputs and compute softmax to get probability distribution over both candidate endings.

$$\begin{bmatrix} P_B\big(y = 1|S, e_1\big) \\ P_B\big(y = 1|S, e_2\big) \end{bmatrix} = \mathrm{softmax}\,\Big(\begin{bmatrix} z_1 \\ z_2 \end{bmatrix}\Big)$$

To obtain final prediction, we take $\arg\max_{i\in\{1,2\}}\ P_B\big(y = 1|S, e_i\big)$.

3.1.4 Additional fine-tuning on masked LM

Instead of fine-tuning BERT only on the final classification task, we can make use of the available large training set, and fine-tune BERT on a "masked language model" and "next sentence prediction" objective first. [2]

3.1.5 Negative random sampling

Since the training data does not include negative candidate endings, we can not use it directly in the absolute/relative classification pipelines described above, as we would be likely to overfit to the positive class. To get around this issue, we attached a random ending to each story in the training set and labeled it as negative.

3.2 Sentiment

Following the approach presented in (Chen et al., 2018), we pre-train a sentiment model on the ROCStories training dataset. To grasp the sentiment content, we embed each sentence s_i in a 3-dimensional vector E_i, using VADER (Hutto and Gilbert, 2014), a rule-based sentiment analysis tool originally meant for sentiment analysis of social media text. [3]

The model consists of an LSTM neural network, with a softmax layer on top, that outputs the predicted sentiment vector for the last sentence E_p using the last hidden state h_4

$$h_i = LSTM(E_i, h_{i-1}), i \in [1, 4]$$
$$E_p = softmax(W_e h_4 + b_e)$$

where W_e, b_e and the LSTM weights are parameters to be learned.

[1] `https://tfhub.dev/google/bert_uncased_L-12_H-768_A-12/1`

[2] To this end, we used `create_pretraining_data.py` and `run_pretraining.py` available at `https://github.com/google-research/bert`, and run the pretraining for 10000 training steps.

[3] Compound component of Vader's embedding was omitted for purpose of this work.

The sentiment model is trained by maximizing

$$sim(S) = \frac{E_p \cdot E_5}{\|E_p\|_2 \cdot \|E_5\|_2},$$

that is the cosine similarity between the true sentiment vector E_5 and the predicted sentiment vector E_p for the ending.

To fine-tune the model for the Story Cloze task we use validation set with two candiate endings. Probability that e_i is the correct ending is here modeled as

$$P_S(y = 1|S, e_1) = softmax(E_p W_s E_e),$$

where $W_s \in \mathbb{R}^{3\times 3}$ is a similarity matrix to be learned and E_e is the sentiment vector extracted from ending e_i.

3.3 Common Sense

Reproducing the approach of (Chen et al., 2018) we use a numberbatch word embedding to include common sense knowledge in our model. This embedding has been trained on data from OpenSubtitles, GloVe, ConceptNet, and word2vec (Speer et al., 2017). First, we remove stopwords and tokenize all sentences using the NLTK toolkit. Subsequently, we calculate a distance vector between each candidate ending and its story body. This is done by calculating the cosine similarity of the vector space representations of the key words. Denote distance vector for (S, e_i) as $D_i \in \mathbb{R}^4$. Classification model on top can again be represented as

$$\underset{i\in\{1,2\}}{argmax} P_C(y = 1|S, e_i) = \underset{i\in\{1,2\}}{argmax}\ \ softmax(W_C^T D_i + b_C),$$

where $W_C \in \mathbb{R}^4$.

3.4 Combining different features

We combine BERT relative classification, common sense and sentiment model by multiplying the three obtained probabilities, i.e. $P = P_S \times P_C \times P_B$.

4 Training

For the sake of the project several datasets were provided.

S I Set of training stories, containing 88161 stories with only the correct endings.

S II Set of training-validation stories, containing 1871 stories with two labeled candidate endings, one of which is correct and one incorrect. Both endings are semantically connected to the story context S.

S III Set of labeled test stories, containing 1871 stories in the same format as previous training-validation stories. Model performance is calculated on this set.

S IV Set of unlabeled test stories, containing 2343 stories with two candidate endings. Submitted results will contain label of the correct ending $\{1, 2\}$ as predicted by our model.

Multiple training sets were constructed in order train the model. For the sake of this report, we name and analyse four of them, which are most relevant:

TS I Set of training stories with negative sampling, consisting of 88161 stories. This set was trained for 1 epoch.

TS II Set of concatenated training stories with random negative sampling and three repetitions of training-validation stories, consisting of $88161 + 3 \cdot 1871 = 93774$ stories, This training set was trained for 1 epoch.

TS III Set of concatenated 1871 training and 1871 training-validation stories. This set is shuffled and trained for 3 epoch.

TS IV Set of training-validation stories, consisting 1871 stories. This set was trained for 3 epochs.

We have trained our models on Google Colaboratory, which offers computing services of Tesla K80 GPU, on the aformentioned training sets for 3(1) epochs, batch size 8, learning rate 0.00002, maximum sequence length 400 and warm-up proportion 0.1. Model hyperparameters were selected manually, our results have shown that they do not affect the model performance significantly, therefore, no cross-validation and hyperparameter tuning was necessary. Sentiment model was trained as a standalone model, pre-trained for 3 epochs on **S I** and fine-tuned for 1 epoch on **S II**.

5 Experiments

Several experiments were conducted to analyze the impact of pre-trained BERT weights, random negative sampling and incorporation of additional information, such as sentiment and common sense. In table 1 we report the accuracies for different models obtained on the labeled test set (**S III**) of the Story Cloze task.

Model	Accuracy
BERT RC **TS III**	0.8809
BERT RC **TS IV**	0.8621
BERT AC **TS III**	0.8581
BERT AC **TS III** + MLM	0.8506
BERT RC **TS II**	0.7988
BERT RC **TS I**	0.5767

Model	Accuracy
Sentiment	0.5874
Common Sense **TS III**	0.5284
Combination BERT RC **TS III**	0.8761

Table 1: Table of models with reported accuracies, where abbreviations mean: classification type - relative (RC) or absolute (AC), training set used TS, MLM - BERT initialization after masked language model pre-training. Sentiment was trained as a standalone model.

6 Conclusion

We demonstrate, that BERT can be effectively applied to the Story Cloze Task and alone achieves state-of-the-art performance. The results showed, that relative classification setting and training on **TS III** achieves best performance, since its training objective is more in line with the discriminating between two candidate endings.

Training on a significantly smaller set **TS III** outperforms training on **TS I** and **TS II**, since samples in **S II** contain endings, which are both semantically connected to the context. Moreover, since BERT is already pre-trained on a large text corpus, it should generalize well to the whole English language. Therefore, training on **TS I** is not necessary and our results showed, that BERT can be effectively fine-tuned to Story Cloze Task on a rather small, but well-structured, **S II**. Both sentiment and common sense models achieve significantly worse performance, where common sense is close to random, therefore, the combination model does not outperform the plain BERT model. We note that low score obtained on common sense model is not aligned with findings from Chen et al. (2018). Narrative model is the most important model part and, as implied by the results, it has sufficient capacity to learn semantical information.

References

Chen, J., Chen, J., and Yu, Z. (2018). Incorporating structured commonsense knowledge in story completion. *arXiv preprint arXiv:1811.00625*.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Hutto, C. J. and Gilbert, E. (2014). Vader: A parsimonious rule-based model for sentiment analysis of social media text. In *Eighth international AAAI conference on weblogs and social media*.

Mostafazadeh, N., Chambers, N., He, X., Parikh, D., Batra, D., Vanderwende, L., Kohli, P., and Allen, J. (2016). A corpus and evaluation framework for deeper understanding of commonsense stories. *arXiv preprint arXiv:1604.01696*.

Mostafazadeh, N., Roth, M., Louis, A., Chambers, N., and Allen, J. (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*, pages 46–51.

Radford, A., Narasimhan, K., Salimans, T., and Sutskever, I. (2018). Improving language understanding by generative pre-training. *URL https://s3-us-west-2. amazonaws. com/openai-assets/research-covers/languageunsupervised/language understanding paper. pdf*.

Speer, R., Chin, J., and Havasi, C. (2017). Conceptnet 5.5: An open multilingual graph of general knowledge. In *Thirty-First AAAI Conference on Artificial Intelligence*.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.