STORY CLOZE TASK

Abhinav Aggarwal
Department of Computer Science
University of Zurich
abhinav.aggarwal@uzh.ch

Ankush Panwar
Department of Computer Science
University of Zurich
ankush.panwar@uzh.ch

July 10, 2020

1 Introduction

Reasoning and logical induction have generated a lot of interest recently in the field of natural language understanding. Narrative comprehension can pave way to enable machines to understand narrative and hold conversations with humans, touted as the first step towards real intelligence. In this project we focus on the Story Cloze task which requires the model to choose the correct ending from two candidates to a four-sentence story. Successful narrative understanding (getting closer to human performance of 100%) requires the model to link various levels of semantics to commonsense knowledge. This issue is particularly challenging for understanding casuality and correlations between events. An interesting feature of this dataset is that the training and validation sets are fundamentally different in the regard that there are no negative example endings for the training set. Surprisingly previous work has shown that ignoring the training set and training a model on the validation set can achieve high accuracy on this task due to stylistic differences between the story endings in the validation and test sets [1], [2]. Our working code can be found at `https://gitlab.com/abhinavaggrwal/nlp2_project`

2 Dataset

We are using ROC Stories (2016) dataset to evaluate our Story Cloze Task which is publically available. To enable the Story Cloze Test, it has a new corpus of five-sentence commonsense stories. This corpus is unique in two ways:

- It captures a rich set of causal and temporal commonsense relations between daily events.
- It is a high quality collection of everyday life stories that can also be used for story generation.

The ROC training dataset contains 65,534 rich stories with only correct ending. However, an evaluation set and a test set contains 4 sentences long stories with 2 candidate ending out of which one is correct (example given in Fig 1) and each has a size of 1,871 stories.

Context short story	**Right Ending**	**Wrong Ending**
Gina misplaced her phone at her grandparents. *It wasn't anywhere in the living room.* *She realized she was in the car before.* *She grabbed her dad's keys and ran outside.*	*She found her phone in the car.*	*She didn't want her phone anymore.*

Figure 1: Validation/ Test story example from ROCStories

3 Methodology & Model

We extract multiple types of information from the stories and train a model on each type. Then, we use a combination gate to integrate all the information and train the model in an end to end manner following the architecture of [3]. The following subsections elaborate on each step of the pipeline.

3.1 Features

We focus on a discriminative model. For each story, we extract three types of information: Language model features, sentiment evolution and commonsense knowledge. Using these information we predict the true ending.

- **Language Model Features:** We use BERT (Bidirectional Encoder Representations from Transformers) introduced by Devlin et al. [4] to extract the embeddings for each story-ending combination. These embeddings contain the information about narrative sequence and entailment of ending given four sentence story
- **Sentiment Evolution:** We use VADER[5] to extract the sentiment scores of the each sentence in story and predict ending sentiment using BiLSTM[6] training.
- **Commonsense Knowledge:** We leverage the implicit knowledge by using a numberbatch word embedding (Speer, Chin, and Havasi 2017) [7], which is trained on data from ConceptNet, word2vec, GloVe, and OpenSubtitles.

Following subsections explain the generation of different features and models in detail.

3.1.1 Language Model Features

The Story Cloze task is special in its design as it has only a limited number of hand selected sentences in its validation set but still requires deep language understanding. As explained in the original paper [8], a shallow understanding of the semantics of the sentences leads to only mediocre results. An effective way to circumvent the problem of limited data and a method to improve deep language understanding and commonsense reasoning is to split the task in two steps. First a large set of unlabeled training data is used to pre-train a generic model, which has no task-specific architecture included, yet. In the second step, the pre-trained generic model is joined with a downstream task specific architecture. This additional sub-model has only a minimal set of additional parameters, as ideally the deeper language understanding should have been accomplished in the general part of the model.

Body	Correct Ending	Wrong Ending
Agatha had always wanted pet birds. So one day she purchased two pet finches. Soon she couldnt stand their constant noise. And even worse was their constant mess.	Agatha decided to return them.	Agatha decided to buy two more.

Table 1: ROC Story example showing break in flow of narrative if wrong ending is taken into consideration. Here in this example if Agatha could not stand the constant noise of pet birds then she should ideally return it and will definitely not buy two more.

As seen in Table 1, to describe a consistent story, plots should be planned in a logically reasonable sequence; that is there should be a narrative chain between different characters in the story. Here if Agatha could not stand the constant noise of pet birds then she should ideally return it and will definitely not buy two more. Hence, having deeper understanding of correct narrative and language semantics is really important to identify the correct ending. As shown in recent research [9], high capacity language models can understand language semantics really well. Therefore we choose BERT to create these semantic representations which can be have deeper understanding of correct narrative.

For this project, we use Huggingface's BERT implementation[10] in pytorch. We load pretrained **BERT**$_{BASE}$ (L=12, H=768, A=12, Total Parameters=110M) uncased model and fine-tuned it for correct ending prediction task. Formally, for each candidate story $(s_1, s_2, s_3, s_4, e_i)$ (i.e., the story body followed by one candidate ending), we serialize it into a sequence of two sentences with concatenated s_1, s_2, s_3, s_4 as first sentence and e_i as second sentence and fine-tune it for story cloze task. Since BERT is trained on next sentence prediction task which is very similar to our task, hence feeding each candidate story as two sentences is natural choice of input preprocessing. The fine-tuned transformer takes both candidate stories as its input and outputs the probability of each ending e_i being the correct ending:

$$P_l(e_i = correct\_ending|s_1, s_2, s_3, s_4, e_i) = softmax(W_l h_l^i + b_l) \tag{1}$$

where h_l is output embedding from `[CLS]` token position at last layer of BERT network, W_l and b_l are parameters of linear layer mapping h_l to single value logit

3.1.2 Sentiment Evolution

Sentiment analysis can provide a crucial insight into the mood (happy, sad, etc.) built up the story so far which is essential to predict the ending. This method is based on following the emotional trajectory of the story as introduced by Chaturvedi et al. [11]. We build on the hypothesis that every story can be divided into the following 4 narrative-segments: a beginning, a body, a climax, and an ending. Thus, we assign sentence 1 as the beginning, sentence 2 and 3 as body, sentence 4 as climax and the sentence 5 [ending 1, 2] as the ending for training [validation] set. The context of the story is defined by the first 4 sentences.

Body	Correct Ending	Wrong Ending
Jackson had always wanted to grow a beard. His friends told him that a beard would look bad, but he ignored them. Jackson didn't shave for a month and he grew a bushy, thick beard. Admiring himself in the mirror, Jackson felt satisfied.	He was glad that he hadn't listened to his friends	He was ashamed of himself.

Table 2: ROC Story example showing consistent flow of sentiment in case of correct ending whereas there is change in sentiment polarity for wrong ending

As seen in Table 2, the emotions between the two candidate endings are different. Based on the rule of consistent sentiment evolution, an appropriate ending should have a positive emotion rather than a negative emotion as ending sentiment should follow the sentiment journey of context.

We pre-train a sentiment prediction model using the training set of the ROCStories, which does not have alternative endings (i.e., no negative samples). Given a five- sentence story $S = s_1, s_2, s_3, s_4, s_5$, we take the first four sentences as the body B and the last sentence as the ending e. Then, we use the following approach to map the sentiment journey of body and use it to predict the ending sentiment.

1. We extract the sentiment polarity of each sentence by utilizing a lexicon and rule-based sentiment analysis tool (VADER) developed by Hutto and Gilbert.
$$E_i = VADER(s_i) \tag{2}$$
2. Then we use biLSTM by taking extracted VADER embeddings as input and predict the sentiment embedding of ending e
$$h_i = LSTM(E_i, h_{i1}), i \in [1, 4] \tag{3}$$
$$E_p = softmax(W_s h_4 + b_s) \tag{4}$$
3. Finally this network is optimized using cosine distance loss between E_p and E_5

At last, we fine-tune the pretrained network on validation set by computing the cosine similarity between E_p and E_i and apply softmax layer to get final probabilities

$$P_s(e_i = correct\_ending|s_1, s_2, s_3, s_4, e_i) = softmax(cos(E_p, E_i)) \tag{5}$$

where e_i is the candidate ending of given context story.

3.1.3 Commonsense Knowledge

Language Model and Sentiment features are useful to predict the correct ending, however they are not sufficient. Since stories in the dataset are short, there are words whose context could not be establish in only five sentences. So, there are many hidden relations among words such as "diet" and "overweight" which could not be interpreted directly using word embeddings. As seen in example 3 and Fig 2, word "Dog" and "find" in correct ending is related to words "Owner's", "Collar" ,"Park" and "stray" in Conceptnet (Commonsense Knowledge graph) sub-graph and incorrect ending doesn't have these implicit word-relations with story context. Thus, commonsense knowledge is very useful in predicting such endings where commonsense implicit relations are present.

In order to incorporate commonsense knowledge and leverage the implicit word-relations in our network, we used ConceptNet NumberBatch word embeddings. ConceptNet NumberBatch is trained on ConceptNet, word2vec, GloVe, and OpenSubtitle and achieves good performance on commonsense knowledge task. Let $S = s_1, s_2, s_3, s_4$ are the 4

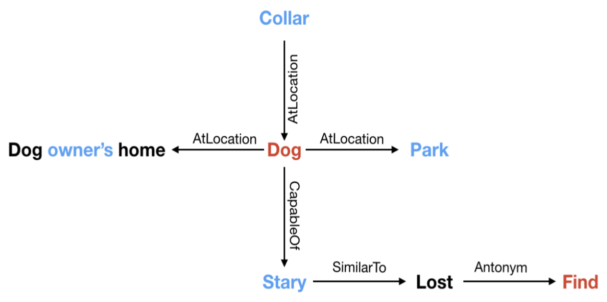

Figure 2: Sub-graph of ConceptNet Numberbatch

Body	Correct Ending	Wrong Ending
I was walking through Central Park on a fall day. I found a stray dog with a collar. I called the number on the collar and talked to the owners. The owners came to the park to pick up their dog.	They thanked me very much for finding their dog	They let me keep it.

Table 3: ROC Story example showing consistent flow of sentiment in case of correct ending whereas there is change in sentiment polarity for wrong ending

sentence story context and e_i be the candidate ending sentenxe. Firslty, we tokenized each sentence after preprocessing the story by removing stop words and then used Numberbatch word-embedding for each word. Then we found cosine similarity of each word in candidate ending with each word of context story sentences s_i and took average alignment score of every key-word in the ending. . It gave us a 4 dimensional distance vector D_i (one element for each ending-context sentence pair. Finally, the D_i vector is passed from a linear layer and softmax to output conditional probability of candidate ending being the correct ending as given in below equation:

$$P_c(e_i = correct\_ending|s_1, s_2, s_3, s_4, e_i) = softmax(W_d D_i + B_d) \tag{6}$$

where W_d and B_d are parameters in the linear output layer, and D_i is the four-dimensional distance vector for the candidate ending.

3.2 Model

In order to create a model using above mentioned features, we first pre-train individual models for three type of features and then use these pretrained models to combine them using a combination gate as described in [3] and fine-tuning the complete model in end-to-end manner. Our model can be seen in Figure 3

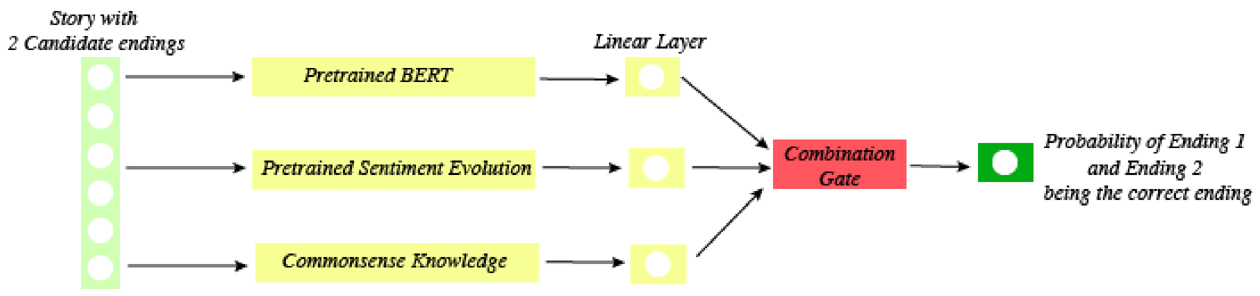

Figure 3: Our Model Architecture implemented to solve the Story Cloze Task.

Training of individual models can be seen in section 3.1.1, 3.1.2 and 3.1.3. Now in next section we will mention in detail about combination gate and end-to-end training procedure.

3.2.1 Combination Gate

In our model with the combination gate, we use embeddings from all three pretrained models for both ending1 and ending2 to dynamically compute the weights. The key motivation behind dynamic weight computation is that the model with more distant embeddings for both endings are more distinctive and thus should be given more weight. Specifically we use hidden representation h_l^i from bert model, E_i from sentiment model and D_i from Commonsense model. We calculate the cosine similarity between h_l^1 & h_l^2 and repeat this process for sentiment and commonsense models as well. Then these similarities are concatenate them into a vector g. Then we pass the cosine similarities to a dense layer, and apply the softmax function to get weighting coefficients for each model. Formally, we do the following:

$$G = softmax(W_g g + b_g) \tag{7}$$

where W_g and b_g are parameters of linear layer and G is vector containing the weighting coefficients. Finally these weights are multiplied with individual model probabilities to get the final prediction.

$$\mathcal{P}(e_i = correct\_ending|s_1, s_2, s_3, s_4, e_i) = sum(G \odot [P_l; P_s; P_c]) \tag{8}$$

where $\odot$ is elementwise multiplication, P_l, P_s, P_c are predicted probabilities for language model, sentiment model and commonsense model respectively and $\mathcal{P}$ is final predicted probability.

4 Training

In this section we mention the training details and hyperparameter values used for training our network. For fine-tuning on story cloze task we only use validation data. 10% of this data is used for validation and remaining 90% is used for training. All the models are trained with Adam optimizer and standard binary cross entropy loss.

- **BERT Model:** We initialize the weights of this model with pretrained **BERT**$_{base}$ uncased model with L=12, H=768, A=12. Then this model is fine-tuned for story cloze task with learning rate of 10^{-5}, batch size of 32 and 3 epochs.
- **Sentiment Model:** For training ending sentiment prediction model, we use training data and train it for 5 epochs with batch size of 64 and 0.001 learning rate. Here LSTM model has single layer with hidden state size of 128. This pretrained model is then fine-tuned on the story cloze task with 10^{-5} learning rate, 2 epochs with batch size of 32.
- **Commonsense Model:** This is simple multi-layer perceptron model with 1 hidden layer with size of 256. It is trained for 5 epochs with batch size of 64 and learning rate of 0.0001
- **Combined Model:** This network is initialized with all pre-trained individual models. Then it is fine-tuned for 3 epochs with 10^{-5} learning rate and 32 batch size

5 Experiments

Even though we have a training set of 65,534 data points, we cannot use it, as the training set only contains correct endings and generating appropriate wrong ending which contains the context of story is a complex as well as difficult task. Therefore we only use ROC validation set,which has 1871 data points, for training the network. We split the validation set such that we can train on 90% of the validation data, and validate on 10%. However complete ROC training(65,534 data points) set is used only to pre-train the sentiment evolution. We report all the results on separate test set.

First of all, we pretrain all 3 models i.e. BERT Language Model, Sentiment evolution and commonsense knowledge network separately and probability of each candidate ending being correct is computed. The resulting accuracy from these models is mentioned in table 5. Secondly, we ensemble the models and feed to the combination gate to allocate appropriate weights to outputs from 3 separate networks. Finally, end-to-end training is performed on the combined-ensemble network of these pretrained models to predict the correct ending. As shown in 4, our model is able to beat all the baseline models where transformer based architecture is not used. Our ensemble model is able to beat most of the transformer based models [12],[3]. Current state-of-the-art has used external dataset to further fine-tune the BERT based transformer model which lead to such significant boost in prediction accuracy. We refrain from using external data to do fair comparison with other transformer based models

Model Type	Method	Accuracy (%)
Baselines	Msap [Schwartz *et al.*2017][1]	75.2
	HCM [Chaturvedi *et al.*2017][11]	77.6
	HBiLSTM [Cai *et al.*2017][13]	74.7
	SeqMANN [Li *et al.*2018][14]	84.7
Transformer based	FTLM [Radford *et al.*2018][12]	86.5
	ISCK [Chen *et al.*2018][3]	87.6
	Trans-Bert [Zhongyang *et al.*2019][15]	91.8
	Our Combined model	**87.8**

Table 4: Experimental results and differences between our best model and all baseline models on test dataset

Our Model	Accuracy (%)
BERT Model	86.7
Sentiment Evolution	56.22
Commonsense Knowledge	57.29
Combined Model (with Combination Gate)	87.8

Table 5: Performance of our model compared to the our sub-models

6 Conclusion

We combined multiple approaches that were already known to perform well on the Story Cloze task, and achieved a higher accuracy. In short as shown in our work, even combining weak ensemble models can lead to significant boost in overall predictive performance of the network. As a future work, one can try to find new and more effective ways to combine these approaches. Another possibility would be to include more stylized features like (i) length (number of words) in the ending sentence, (ii) negation that is presence of words like none, never, etc. in the ending (iii) word n-grams (iv) character n-grams.

References

[1] R. Schwartz, M. Sap, I. Konstas, L. Zilles, Y. Choi, and N. A. Smith, "Story cloze task: UW NLP system," in *Proceedings of the 2nd Workshop on Linking Models of Lexical, Sentential and Discourse-level Semantics, LSDSem@EACL 2017, Valencia, Spain, April 3, 2017*, M. Roth, N. Mostafazadeh, N. Chambers, and A. Louis, Eds. Association for Computational Linguistics, 2017, pp. 52–55. [Online]. Available: https://doi.org/10.18653/v1/w17-0907

[2] S. Srinivasan, R. Arora, and M. Riedl, "A simple and effective approach to the story cloze test," 03 2018.

[3] J. Chen, J. Chen, and Z. Yu, "Incorporating Structured Commonsense Knowledge in Story Completion," *arXiv e-prints*, p. arXiv:1811.00625, Nov. 2018.

[4] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," *arXiv e-prints*, p. arXiv:1810.04805, Oct. 2018.

[5] C. Hutto and E. Gilbert, "Vader: A parsimonious rule-based model for sentiment analysis of social media text," 01 2015.

[6] Z. Huang, W. Xu, and K. Yu, "Bidirectional LSTM-CRF Models for Sequence Tagging," *arXiv e-prints*, p. arXiv:1508.01991, Aug. 2015.

[7] R. Speer, J. Chin, and C. Havasi, "ConceptNet 5.5: An open multilingual graph of general knowledge," 2017, pp. 4444–4451. [Online]. Available: http://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14972

[8] N. Mostafazadeh, N. Chambers, X. He, D. Parikh, D. Batra, L. Vanderwende, P. Kohli, and J. Allen, "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*. San Diego, California: Association for Computational Linguistics, Jun. 2016, pp. 839–849. [Online]. Available: https://www.aclweb.org/anthology/N16-1098

[9] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," 2019.

[10] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, and J. Brew, "Huggingface's transformers: State-of-the-art natural language processing," *ArXiv*, vol. abs/1910.03771, 2019.

[11] S. Chaturvedi, H. Peng, and D. Roth, "Story comprehension for predicting what happens next," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Copenhagen, Denmark: Association for Computational Linguistics, Sep. 2017, pp. 1603–1614. [Online]. Available: https://www.aclweb.org/anthology/D17-1168

[12] A. Radford, "Improving language understanding by generative pre-training," 2018.

[13] Z. Cai, L. Tu, and K. Gimpel, "Pay attention to the ending:strong neural baselines for the ROC story cloze task," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Vancouver, Canada: Association for Computational Linguistics, Jul. 2017, pp. 616–622. [Online]. Available: https://www.aclweb.org/anthology/P17-2097

[14] Q. Li, Z. Li, J.-M. Wei, Y. Gu, A. Jatowt, and Z. Yang, "A multi-attention based neural network with external knowledge for story ending predicting task," in *Proceedings of the 27th International Conference on Computational Linguistics*. Santa Fe, New Mexico, USA: Association for Computational Linguistics, Aug. 2018, pp. 1754–1762. [Online]. Available: https://www.aclweb.org/anthology/C18-1149

[15] Z. Li, X. Ding, and T. Liu, "Story Ending Prediction by Transferable BERT," *arXiv e-prints*, p. arXiv:1905.07504, May 2019.