Story Completion with Explicit Modeling of Commonsense Knowledge

Mingda Zhang   Keren Ye   Rebecca Hwa   Adriana Kovashka
Department of Computer Science
University of Pittsburgh
{mzhang, yekeren, hwa, kovashka}@cs.pitt.edu

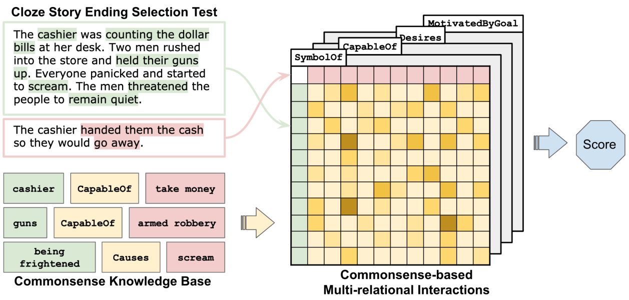
Abstract

Growing up with bedtime tales, even children could easily tell how a story should develop; but selecting a coherent and reasonable ending for a story is still not easy for machines. To successfully choose an ending requires not only detailed analysis of the context, but also applying commonsense reasoning and basic knowledge. Previous work [8] has shown that language models trained on very large corpora could capture common sense in an implicit and hard-to-interpret way. We explore another direction and present a novel method that explicitly incorporates commonsense knowledge from a structured dataset [11], and demonstrate the potential for improving story completion.

1. Introduction

Understanding how a story develops is a natural human skill, and even children understand basic story structures with limited training; numerous kids' educational activities exist for story ending prediction. However, although modern AI models have achieved impressive performance in many fundamental vision and linguistic tasks, predicting the ending for a given story context is still challenging. In fact, story completion usually requires more information than what is available from the literal story context and ending expressions, and much of this knowledge comes from daily experience, also known as "common sense".

In this project, we explore the opportunity of directly injecting commonsense knowledge to help machines to tackle the Story cloze ending selection test [7]. Inspired by how humans reason, we emphasize that not only the *story coherency* should be taken into consideration, but the underlying interactions i.e. *relational knowledge* are also important. Specifically, our proposed method incorporates a commonsense knowledge reasoning module with a standard language model to enhance the ending prediction. The language module handles narrative coherency using a Bidirectional LSTM model, and the knowledge module handles multi-relational interactions among concepts within the context and endings, which is inferred by a model trained from an external commonsense knowledge database.



The cashier was counting the dollar bills at her desk. Two men rushed into the store and held their guns up. Everyone panicked and started to scream. The men threatened the people to remain quiet.

The cashier handed them the cash so they would go away.

Figure 1. A relation classifier is first trained on commonsense knowledge database as a generalizable scorer, which is later used to evaluate the relations between concepts within context and ending. The resulting 3-d tensor contains pairwise interaction across multiple relations. An attention-based aggregation strategy is used to generate the final score.

The major difference between our work and previous ones is that we do not rely on *implicit* commonsense knowledge which were obtained as a by-product from language modeling on a large corpus. Instead, we aim to *explicitly* inject common sense from well-structured knowledge base, thus the results may be easier for human interpretation.

2. Related work

2.1. Story cloze test

Understanding narrative structure is especially challenging when commonsense knowledge is involved. The ROCStories dataset [7] provides an opportunity as it contains crowd-sourced stories consisting of four-sentence context and alternative endings. This dataset features rich topic diversity and causal/temporal commonsense relations, thus has attracted much attention from the community.

For example, [9] took neural language modeling features together with linguistic stylistic features and set up a strong baseline even with simple linear classifiers. [2] built an attention-enhanced hierarchical recurrent neural network, showing the effectiveness of neural language models. A breakthrough came from [8], which built a "pretraining" language model on a large unlabeled corpus. With minimal "finetuning", their model achieved superior performance over a series of natural language understanding tasks including the story ending prediction.

These findings agree with our observation that leveraging external knowledge is critical for solving the commonsense reasoning task. However, we also emphasize that their model acquires knowledge from pretraining on enormous data, but such knowledge is *hard to interpret* for humans.

Recently, [4] proposed a network on top of [8], adding two additional branches for modeling sentiment evolution and commonsense knowledge. Their approach inspired us to further explore the possibility of injecting structured commonsense knowledge more directly.

2.2. Commonsense knowledge base completion

Most of the knowledge base completion models predict missing relation links between entities. In other words, the problem is formulated as classification over all possible relations. For example, [1] treated the relation as an operation and used a translation model to transform the entities; [10] used a hybrid model which combined a multi-layer perceptron and a bilinear model; [6] resembled our proposed approach the most in terms of relationship prediction. Specifically, we also encode arbitrary entities and build bilinear models to infer missing links in ConceptNet. However, our end goal is very different: we focus on leveraging commonsense cues to measure how well a potential ending matches the story, while they stop at the knowledge base completion.

3. Methodology

Our proposed model contains two separate branches, language modeling and commonsense reasoning. For a given story with two endings, each branch *independently* evaluates the compatibility between context and each ending, and a final decision is achieved by averaging scores from the two branches. In this work more attention is paid to the commonsense reasoning branch.

The core idea of our commonsense module is to first train a relation classifier on a commonsense knowledge base, then use the trained classifier to predict finer-grained pairwise relationships between all concepts in story context and endings. Finally, a compatibility score between ending and context is calculated from all the word-level relationships.

3.1. Generalizable relationship prediction

ConceptNet [11] is a well-structured semantic network with facts and general knowledge encoded daily, providing a valuable resource for adding human knowledge to machine models. In this specialized knowledge graph, words and phrases are linked by labeled, weighted edges, reflecting the relations between concepts. Two examples from ConceptNet are shown below.

- < *story*, USEDFOR, *putting kids to sleep* >
- < *answer question*, REQUIRES, *knowledge* >

However, even with over a million concepts and over 30 million edges among these concepts, ConceptNet is still highly sparse and many plausible entity-relation-entity tuples are missing. For example, <*a friend*, USEDFOR, *making you happy*> exists while <*friend*, USEDFOR, *happiness*> is missing. This drawback prevents some naive usage such as lookup or word matching, since arbitrary phrases can hardly find corresponding entities in ConceptNet.

To address this issue, we develop a generalizable relation classifier from ConceptNet, which takes a pair of *arbitrary phrases* and gives predictions regarding their relationships. Specifically, following [6], we use a BiLSTM model to encode variable-length phrase entities, obtaining fixed-length vector representations. Next, we train a bilinear relationship scoring function g:

$$g(t_1, t_2, r) = f(\text{BiLSTM}(t_1))^T W_r f(\text{BiLSTM}(t_2)) \quad (1)$$

$$f(x) = ReLU(W_v x + b_v) \quad (2)$$

where r represents a specific relation, t_i represents entity (which consists of multiple words) and W_v and b_v represent the weight and bias for linear projection of encoded entity.

Our proposed relation classifier is trained on ConceptNet and achieves comparable performance with the literature [6]. After the training, the model is used off-the-shelf as a scoring function in the downstream tasks.

3.2. Modeling context-ending interactions

Intuitively, if a context has a closer relationship with one ending than the other, then it is more likely to be a reasonable finale for the story. Therefore, our model takes advantage of the knowledge we learned from ConceptNet, and pairwise evaluates the interactions between all the concepts within the context and the ending.

Specifically, we represent the context as a list of concepts $(w_C^1, \cdots, w_C^M)$, and similarly for both ending candidates $(w_{E_i}^1, \cdots, w_{E_i}^N,\ i \in \{1, 2\})$. Next, for each pair w_C^m and $w_{E_i}^n$, we encode the concepts and feed to the aforementioned relationship classifier g, obtaining a vector of R scores, each corresponding to one relation. Therefore, for a context with M words and an ending with N words, we obtain a 3-d tensor $S \in \mathbb{R}^{M \times N \times R}$, where each element represents the score between the m^{th} word in the context and the n^{th} word in the ending, for the r^{th} relation.

We do the same calculation for both endings and obtain two separate tensors, then we aggregate these tensors to obtain a single scalar value to represent the compatibility of the ending with the context. Note that once the relationship predictor is trained on the ConceptNet corpus, the resultant tensor is fully deterministic. Therefore, only the aggregation part is trained on the ROCStories dataset.

3.3. Scoring by aggregation

Given the 3-d tensor, the next step is to aggregate values from the tensor and obtain a scalar compatibility score for the context and this ending. We adapted different strategies to aggregate over relations and over words.

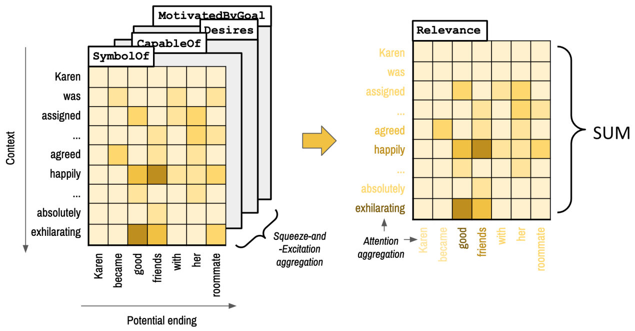


Figure 2. Figurative illustration of the aggregation procedure of the relation tensor data. Rows indicate context words, columns indicate ending words, and relations are described by different x-y planes. The *aggregation over relations* (left) scales different relations for all word-pairs, and the *aggregation over words* (right) assigns a weight to each word pair using attention mechanism, followed by a weighted summation.

Aggregation over relations In Figure 2, the z-axis represents different relationships, as two phrases can be relevant in various ways. From experiments a "squeeze-and-excitation" mechanism [5] can dynamically alter the score across relations and achieve the best performance.

Specifically, the 3-d tensor is first average-pooled on the x-y plane, obtaining $\mathbf{z} \in \mathbb{R}^R$. This is a rough estimation of the relative activation strength for different relations. Then a two-layer MLP is applied to predict a suitable scaling factor $\mathbf{s} \in \mathbb{R}^R$ for different relations, similar with [5]. Next the relational scaling factor is multiplied with the initial tensor to adjust the unbalanced activation across different relations, then reducing to $\tilde{S} \in \mathbb{R}^{M \times N}$.

Aggregation over words For the 2-d matrix $\tilde{S}$, the scores represent pair-wise interactions of all words in the context and ending. We adapt an attention mechanism to dynamically alter the weights on different words.

$$V_C = \frac{1}{M} \sum v_{w_C^m}, \alpha_C^m = \text{softmax}(v_{w_C^m}^T W_w^{attn} V_C^*) \quad (3)$$

$$V_{E_i} = \frac{1}{N} \sum v_{w_{E_i}^n}, \alpha_{E_i}^n = \text{softmax}(v_{w_{E_i}^n}^T W_w^{attn} V_{E_i}^*) \quad (4)$$

$$S^*[m,n] = \alpha_C^m \cdot \alpha_{E_i}^n \cdot \tilde{S}[m,n], \quad (5)$$

For either context or ending, we average all the individual word representations to obtain a global representation (V_C^* and $V_{E_i}^*$), and build a bilinear model to measure the similarity (also the attention score) of individual words under the context. Next the attention scores are multiplied by the relation scores to reflect the relative importance. S^* is the final score matrix, which is further reduced to a scalar value by average pooling.

3.4. Language Model

While the previous module allows us to adaptively bring in relevant external knowledge, it does not preserve the ordering information. Therefore another important module in our model is the language model branch, which focuses on the coherence of the narrative structures. Specifically, we adapt a BiLSTM-based recurrent neural network to encode the context and endings. After the entire sequence is digested, the resultant vector representations are fed to a bilinear model to give scores for the compatibility.

4. Experiments

Similar to some of the previous studies [2, 4], we randomly split the validation set from ROCStories and use 80% for training. The remaining 20% of the validation set is used for tuning hyperparameters, and the final scores are evaluated on the test set.

4.1. Commonsense relation classification

The foundation for the proposed pipeline is the commonsense relation classifier, as it gives basic prediction of whether two concepts are related, and how they are related.

We followed the experimental setting of [6], and used the same split for a fair comparison.

Our commonsense relation classifier achieved 92.0% overall prediction accuracy on DEV2 set, slightly better than the 91.0% in [6]. Besides, we analyzed the precision and recall, since in our design the positive prediction accuracy (precision) plays a more important role as it serves as the foundation for inference in the next step. Our model obtained 95.3% precision (i.e. accuracy on positives) and 89.4% recall. We also analyzed per-relation prediction accuracy in Figure 3. Specifically, the red bar plot represents the number of samples in the TRAIN set for each category, and the dotted blue line represents the prediction accuracy by our model on DEV2 set. We emphasized that in most categories the accuracy is above 90%, but for some less abundant relations like SYMBOLOF and CREATEDBY, the prediction performance is highly unstable. Note that the least abundant 10 relations even have no test samples in the DEV2 set (no accuracy). To minimize the impact from unreliable predictions, we discard relations that have less than 1000 training samples and low accuracy.

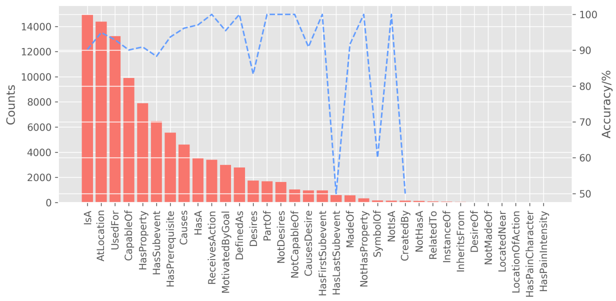


Figure 3. Per-relation analysis. Red bar plot: Number of samples per relation in the TRAIN; Blue dotted line: Accuracy on DEV2.

Our Methods	**Acc(%)**
Ours - Commonsense	64.1
Ours - BiLSTM Language Model	70.1
Ours - Full Model	72.1
Rule-based commonsense with pretraining LM	
[4] - Commonsense	63.8
[4] - Pretraining LM	85.3
[4] - Commonsense + Pretraining LM	87.2
Other approaches w/o pretraining but addl. features	
DSSM [7]	58.5
MSAP [9]	75.2
Cai [2]	74.7
HCM [3]	77.6

Table 1. Performance comparison with other methods.

4.2. Story cloze test

After demonstrating that our relation classifier can predict relations alone, we next show how this classifier helps with the story cloze test.

To validate the effectiveness of our relationship classifier, we compare against several baselines, specifically:

- **Search within ConceptNet** The recovered relations would be highly reliable, but it may suffer from low-coverage. Also since this method does not allow estimation of relative importance for different word-pairs/relations, we rely on the absolute score from ConceptNet and simply average all the scores.
- **Cosine distance on pretrained word embeddings** Similar with [4], we also modified our model to use cosine distance between all words in context and ending (pairwisely) as a baseline. In this case the aggregation over relations would be unnecessary as cosine only gives a single similarity score, but we still aggregate over words with the same proposed strategy.

From our experiments, directly searching only gets 48.1% accuracy. We suspect the low performance is due to the missing entries, as the prediction is almost purely relying on the "hit" rate in knowledge base. The cosine similarity captures certain connections and the accuracy improves to 57.0%. Finally, **our proposed per-relation scoring strategy achieves the highest accuracy of 64.1%.**

We also show that **our commonsense module** slightly outperforms [4]. It is worth noting that in their approach the context is summarized as 4-d vector by a rule-based algorithm using cosine distance, thus the results are hard to interpret. However, for our module, since the decision is based on S^*, it is feasible to trace all the way back to identify the word pairs (and relation) with strongest interactions. [4] also use Transformer-based language model (pretrained on 7000+ books), thus results are not directly comparable.

In Table 1, we see that the **incorporation of our knowledge branch improves** upon the language-only branch. The improvement is consistent with [4] in which they use a massively pretrained language model. In addition, we observe that our method achieves comparable results to several prior methods. Note that these contain a variety of features in combination, e.g. [3] feature information about events, topics, and sentiments. In contrast, our full method only relies on commonsense knowledge and a BiLSTM language model, and requires less human curation.

5. Conclusion

In this paper, we presented an approach that leverages explicit external knowledge for the story completion task. We trained a relationship predictor for a pair of arbitrary entities, then applied this predictor to all pairs of words from the story context and each possible ending. We demonstrated our knowledge branch improves upon a language-only model and achieves comparable performance to a prior knowledge-only method. In the future, we will further leverage this interpretability by performing multi-hop reasoning and finding chains of relation types to explain the connection between context and story ending, and apply the module to cross-modality commonsense reasoning.

References

[1] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In *NeurIPS*, 2013.

[2] Zheng Cai, Lifu Tu, and Kevin Gimpel. Pay attention to the ending: Strong neural baselines for the roc story cloze task. In *ACL*, 2017.

[3] Snigdha Chaturvedi, Haoruo Peng, and Dan Roth. Story comprehension for predicting what happens next. In *EMNLP*, 2017.

[4] Jiaao Chen and Zhenglin Yu. Incorporating structured commonsense knowledge in story completion. In *AAAI*, 2019.

[5] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *CVPR*, 2018.

[6] Xiang Li, Aynaz Taheri, Lifu Tu, and Kevin Gimpel. Commonsense knowledge base completion. In *ACL*, 2016.

[7] Nasrin Mostafazadeh, Nathanael Chambers, Xiaodong He, Devi Parikh, Dhruv Batra, Lucy Vanderwende, Pushmeet Kohli, and James Allen. A corpus and cloze evaluation for deeper understanding of commonsense stories. In *NAACL-HLT*, 2016.

[8] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. In *technical report, OpenAI*, 2018.

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

[10] Richard Socher, Danqi Chen, Christopher D Manning, and Andrew Ng. Reasoning with neural tensor networks for knowledge base completion. In *NeurIPS*, 2013.

[11] Robyn Speer, Joshua Chin, and Catherine Havasi. Conceptnet 5.5: An open multilingual graph of general knowledge. In *AAAI*, 2017.