Integrating External Event Knowledge for Script Learning

Shangwen Lv[1,2], **Fuqing Zhu**[1*], **Songlin Hu**[1,2*]
[1]Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
[2]School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
{lvshangwen,zhufuqing,husonglin@iie.ac.cn}

Abstract

Script learning aims to predict the subsequent event according to the existing event chain. Recent studies focus on event co-occurrence to solve this problem. However, few studies integrate external event knowledge to solve this problem. With our observations, external event knowledge can provide additional knowledge like temporal or causal knowledge for understanding event chain better and predicting the right subsequent event. In this work, we integrate event knowledge from ASER (Activities, States, Events and their Relations) knowledge base to help predict the next event. We propose a new approach consisting of knowledge retrieval stage and knowledge integration stage. In the knowledge retrieval stage, we select relevant external event knowledge from ASER. In the knowledge integration stage, we propose three methods to integrate external knowledge into our model and infer final answers. Experiments on the widely-used Multi-Choice Narrative Cloze (MCNC) task show our approach achieves state-of-the-art performance compared to other methods.

1 Introduction

A script is a sequence of stereotypical events related to a protagonist. A restaurant script about "Jack" can consist of "Jack walked to a restaurant", "Jack read the menu", "Jack ordered food" and "Jack ate food". Script learning aims to predict the subsequent event given the event chain. According to the script above, we can predict the next event could be "Jack payed for the food" or "Jack left restaurant". Script learning can be applied to a wide range of applications, such as storytelling, dialogue generation, discourse understanding and intention recognition, etc.

There have been recent lines focusing on script learning. The first line models the event co-occurrence to infer the next event. Event pair methods like PMI (Chambers and Jurafsky, 2008) and event bi-gram (Jans et al., 2012) aim to model the pair relation between predicted event and events within the chain. Event chain methods like LSTM (Pichotta and Mooney, 2016) model the event chain to predict the next event. PairLSTM (Wang et al., 2017) and SAM-Net (Lv et al., 2019) both utilize event pair and event chain modeling to predict the subsequent event. SGNN (Li et al., 2018) constructs an event graph and utilizes the graph information. The second line goes beyond the event-occurrence and utilizes discourse relations or external commonsense knowledge. (Lee and Goldwasser, 2019) utilizes Trans-*(TransE,TransR,TransH) to learn the relation between events and (Ding et al., 2019) aims to integrate external commonsense knowledge like sentiment and intention into event representations. However, the above studies ignore the effect of external event knowledge, which can provide abundant temporal and causal knowledge for events.

With our observations, external event knowledge plays a significant role in understanding and predicting events. As shown in Figure 1, the solid circles in the line are events in the given chain and the dotted event is the event to predict. The triples in "(head event, relation, tail event)" format are the knowledge from external knowledge bases. When understanding the events in the chain, we can see that the triple

*Corresponding author.

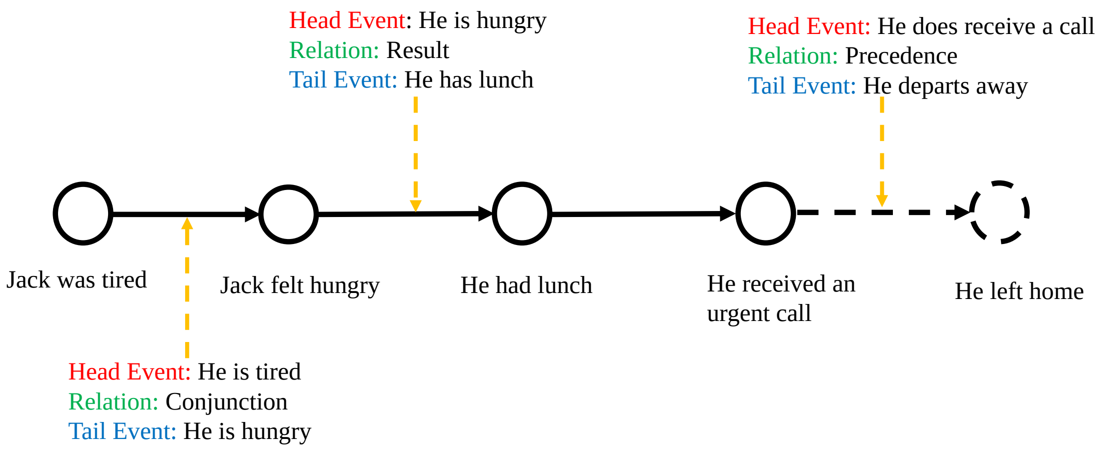

Figure 1: An example of script learning. The solid circles are the events in the given chain and the dotted circle is the event to predict. The triples in (head event, relation, tail event) format are knowledge from external event knowledge bases.

"(He is hungry, Result, He has lunch)" gives evidence to understand that the event "He had lunch" is the result of "Jack felt hungry". When predicting the next event, we can see that the triple "(He do receive a call, Precedence, He depart away)" can provide enough evidence to predict the next event will be "He left home". From the example, we can see that external event knowledge would benefit the script learning problem.

In this work, we propose to utilize external event knowledge to help resolve the script learning problem. Our approach consists of two stages: knowledge retrieval and knowledge integration. In the knowledge retrieval stage, we aim to locate the given individual event within the script in knowledge bases. We utilize Elastic Search to select relevant events and then we design rule-based methods to re-rank the relevant events to obtain the most relevant event. In the knowledge integration stage, we propose three methods including "Tail Only", "Event Template" and "Relation Embedding" to integrate external knowledge sources into our model to help infer the subsequent event.

The contributions of this paper can be summarized as follows:

- To the best of our knowledge, we are the first to integrate external event knowledge for script learning problem, which consists of knowledge retrieval stage and knowledge integration stage.
- We propose three different approaches to integrate external knowledge into our model and compare the effects of three methods.
- Experimental results on the MCNC task show our approach can get state-of-the-art results compared with other methods.

2 Related Work

Script (Schank and Abelson, 2013) is an event sequence related to a protagonist. It is first defined by human-experts to help question answering systems, which is time-consuming and labor-intensive. In (Chambers and Jurafsky, 2008), the researchers propose to utilize statistical models to learn the co-occurrence between events and predict the subsequent event. Many studies follow (Chambers and Jurafsky, 2008) and propose many approaches. Two major approaches are event pair modeling and event chain modeling.

In event pair modeling, researchers adopt event n-gram (Jans et al., 2012) or event pair co-occurrence (Pichotta and Mooney, 2014) to predict the subsequent event. (Granroth-Wilding and Clark, 2016) first introduces word embeddings to represent events and achieves good results on the task. After that, embedding-based methods have been a dominant approach in this area.

https://www.elastic.co/

For event chain approaches, (Pichotta and Mooney, 2016) first utilizes Long Short Term Memory (LSTM) to model the event chain and predict the subsequent event. Many approaches not only consider event chain modeling, but also utilize event pair modeling. (Wang et al., 2017) utilizes event pair modeling and event order into consideration and achieves better performance. (Lv et al., 2019) proposes to use self attention mechanism to discover event segments as event chain modeling and combine event pair modeling to predict event.

Different from the above studies, (Li et al., 2018) constructs an event evolutionary graph to represent the co-occurrence between events and utilize the information from neighbors to predict events.

The aforementioned methods only consider the evet co-occurrence. (Lee and Goldwasser, 2019) goes beyond the event co-occurrence and utilize discourse relations to model the relations between event. (Ding et al., 2019) introduces external sentiment and intention information into events to improve event representations.

In this work, we propose to integrate external event knowledge to help understand and predict the subsequent events, which can provide abundant temporal and causal information about events.

3 Problem Definition

As shown in Figure 2, script learning aims to predict the subsequent event e_{n+1} given the event chain $<e_1, e_2, \cdots, e_n>$. An event e_i consists of four parts: subject e_s, verb e_v, object e_o and prepositional object e_p. We denote an event as "$e_v(e_s, e_o, e_p)$". For example, we can extract the event "gave(Mary, the book, Jack)" from the sentence " Mary gave the book to Jack". If an event do not have one part, it will be denoted as "None".

Following (Granroth-Wilding and Clark, 2016; Wang et al., 2017; Li et al., 2018; Lv et al., 2019), we also adopt the Multi-Choice Narrative Cloze (MCNC) task to evaluate the effectiveness of different models. This task aims to select the right subsequent event from a set of events $\{e_{c_1}, e_{c_2}, \cdots, e_{c_m}\}$, where m is the number of candidate event. Accuracy is adopted as the metric.

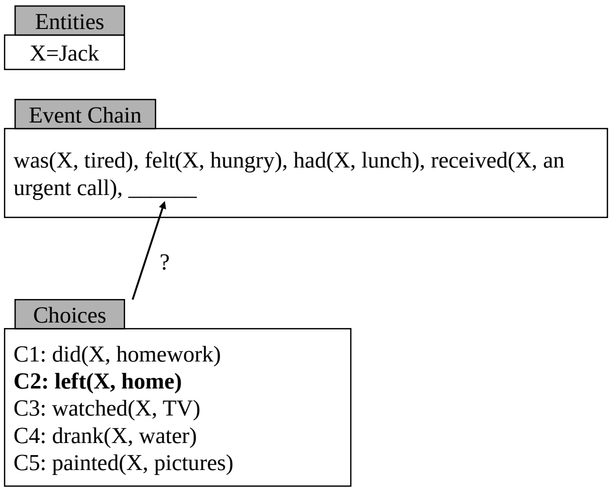

Figure 2: Problem Definition. The script is about the protagonist "Jack".

4 Methodology

Our proposed approach consists of two stages: knowledge retrieval and knowledge integration. In the knowledge retrieval stage, we retrieve relevant knowledge from external event knowledge bases. In the knowledge integration stage, we integrate the retrieved knowledge into our model to infer final answers. We will first give a brief introduction to ASER knowledge base and show the details in the following sections.

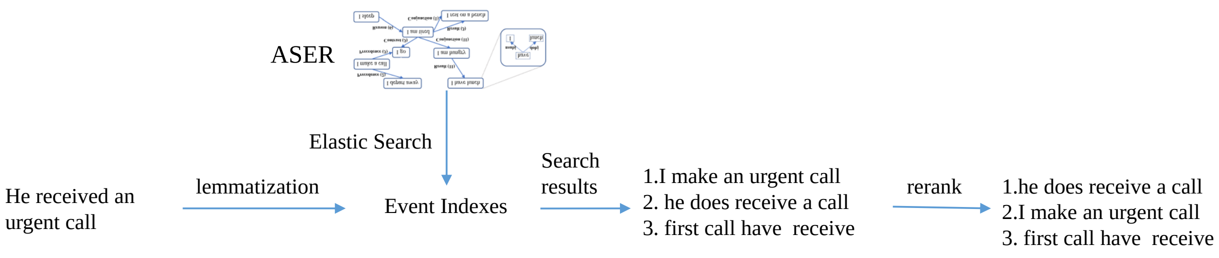

Figure 3: Knowledge Retrieval process. During the lemmatization process, the word "received" will be converted to "receive".

4.1 Brief Introduction to ASER

ASER (Activities, States, Events and their Relations) (Zhang et al., 2020) is an event knowledge base containing 15 event relation, 194 million unique events and 64 million edges among events. The events and relations are extracted from 11-billion-token unstructured text such as Wikipedia, Gigawords, Book-Corppus, etc. The 15 relations reflect the diverse relations like temporal or causal between events. ASER provides abundant knowledge to understand events and is a good treasure to help resolve the script learning problem.

4.2 Knowledge Retrieval

In the knowledge retrieval stage, we aim to select the relevant knowledge about the given event in the script. It can be divided into two parts: locating the event $optim\_event$ in ASER and select the triples in ASER related to $optim\_event$. The first part is similar to entity linking in knowledge graphs. As we can know from the event definition, an event consists of four components, the verb, the subjective, the objective and the prepositional object. It is hard to locate the exact $optim\_event$ in the knowledge bases. In order to relieve this problem, we propose the first-retrieve-then-rerank method. We first retrieve relevant events according to information retrieval methods. And then we rerank the results according to the component of events and obtain $optim\_event$. The process is shown in Figure 3.

There are 194 million events in ASER and it is difficult to obtain $optim\_event$. In order to relive the burden on computation cost, we utilize Elastic Search to construct index for all the events. Elastic Search constructs inverted index for all the events in ASER. When we input one event after lemmatization, the Elastic Search engine will return top K events related to the given event, ranked by the TFIDF or BM25 scores. The search results of the event "He received an urgent call" is shown in Figure 3.

Elastic Search engine utilizes TFIDF and BM25 to retrieve relevant events. However, it regards each component in the event with the same weight. As we know, the verb in an event is more important and expresses more semantics than other components. According to this observation, we design a rule-based score function to calculate the similarities between the given event and the retrieved event. If the verb is the same, we will add 4 scores. If other components are the same, we will add 2 scores. Each event in the event list will be assigned a score and the event with the highest score will be $optim\_event$ according to the given event in the script. In Figure 3, we can select the event "he do receive a call" as $optim\_event$ in ASER. Finally, we select the triples in ASER related to $optim\_event$ as our external event knowledge.

It is worth noting that if the given event does not match any event in ASER, Elastic Search engine will also return an event list. We set a score threshold when selecting the most related event. If the scores are all lower than the threshold. We will retrieve no knowledge for the given event. In our experiments, the threshold is set to 4.

4.3 Knowledge Integration

In this section, we propose three methods to integrate retrieved external event knowledge into our model and compare the effectiveness of different methods.

In our experiments, we set K=10.

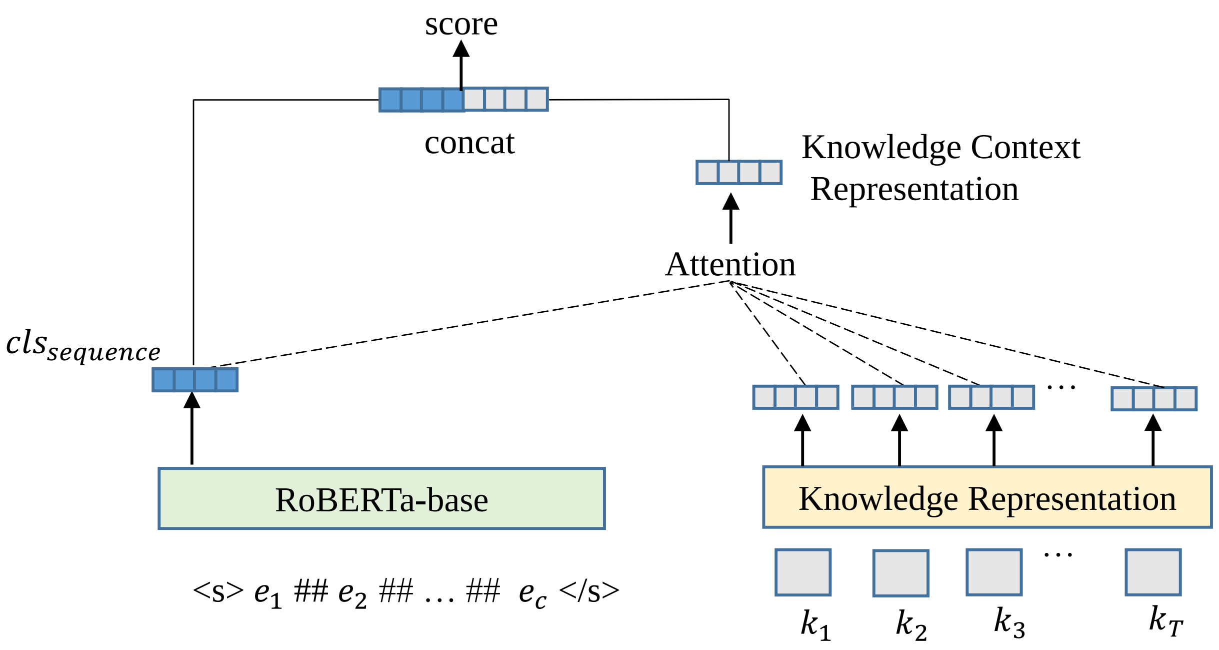

Figure 4: The knowledge integration methods. e_i is the i-th event in the script and e_c is the candidate event. Each event is converted into natural language as "e_s, e_v, e_o, e_p" format. k_i is the i-th triple extracted from ASER and the total number is T. The score is the probability that the model selects the candidate answer. Each canidate event will have a score and the one with the highest score will be selected as the predicted answer.

In recent years, pre-trained models like BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019) have achieved great improvements over a variety of downstream tasks such as question answering, machine reading comprehension, sentiment classification, etc. In this paper, we transfer the pre-trained RoBERTa-base model to model the event sequences. As shown in Figure 4, we put the events in the script and the candidate event into a sequence and utilize "##" as the separator. We also two special tokens <s> and "</s>" which denotes the start and end of the input. The representation of <s> is also known as the <cls> representation. In RoBERTa model, the <cls> is the representation of the whole input and it represents the coherence between the script and the candidate event. We utilize <cls> to interact with external event knowledge to help predict the right answer. We denote the <cls> representation for the input as $\text{cls}_{sequence}$. For simplicity, we denote the knowledge triple as $< h, r, t >$.

We regard retrieved external knowledge as memories and we propose to utilize attention mechanism to integrate external event knowledge into our model. We propose three methods to utilize event knowledge: **Tail-Only**, **Event Templates**, **Representation Fusion**.

For the **Tail Only** method, we only utilize the tail event t in a triple, removing the head event h and relation r. We will first utilize RoBERTa-base to get the representation of the tail event $\text{cls}(t)$ and then we adopt attention mechanism to integrate external knowledge:

$$\begin{aligned}\text{cls}(t_i) &= \text{RoBERTa}(t_i), i = 1, 2, ...k\,,\\ a_i &= \text{softmax}(\text{cls}_{sequence} \cdot \text{cls}(t_i))\,,\\ c &= \sum_{i=0}^{k} a_i \cdot \text{cls}(t_i)\,,\end{aligned} \tag{1}$$

where $\text{cls}_{sequence}$ is the representation of the concatenation of the script and the candidate event. k is the number of external knowledge triples. a_i is the normalization weight over all the triples and $\sum_i^k a_i = 1$. c is the context representation of external knowledge.

For the **Event Template** method, we adopt the templates in ASER to convert the relation r into natural language format. For example, the relation "Precedence" will be converted into "happen before" and the triple “(He does receive a call, Precedence, He departs away)” will be converted into “He does receive a call happen before he departs away”. Then we utilize RoBERTa-base to get the representation of the triple in natural language format and utilize attention mechanism to aggregate external knowledge:

$$\begin{aligned} \text{cls}(triple_i) &= (triple_i), i = 1, 2, ...k\,, \\ a_i &= \text{softmax}(\text{cls}_{sequence} \cdot \text{cls}(triple_i))\,, \\ c &= \sum_{i=0}^{k} a_i \cdot \text{cls}(triple_i)\,, \end{aligned} \tag{2}$$

where $\text{cls}_{sequence}$, a_i and c have similar meanings to those in **Tail Only** method.

For the **Representation Fusion** method, we assign a vector representation for each relation in ASER. Each relation will have the same representation in all the triples containing it. The head event and tail event representation is obtained by RoBERTa-base. Next we fuse the representation of head event, tail event to get the triple representation. For each triple $< h, r, t >$:

$$\begin{aligned} \text{cls}(h_i) &= \text{RoBERTa}(h_i)\ i = 1, 2, ...k\,, \\ \text{cls}(t_i) &= \text{RoBERTa}(t_i)\ i = 1, 2, ...k\,, \\ \text{embedding}(r_i) &= U \cdot \text{relation\_embedding}(r_i)\ i = 1, 2, ...k\,, \\ triple_i &= \text{Linear}([\text{cls}(h_i); \text{cls}(t_i); \text{embedding}(r_i)]\,, \end{aligned} \tag{3}$$

where $\text{relation\_embedding} \in \mathbb{R}^{p*hidden}$, p is the category of relations, $hidden$ is the hidden size. $U \in \mathbb{R}^{p}$ denotes the category of relation, with only one position is 1 and others are 0. We obtain the embedding of r according to its index in the matrix. [;] denotes the concatenation operation. Next, we aggregate the external triple representations to get the context representation:

$$\begin{aligned} a_i &= \text{softmax}(\text{cls}_{sequence} \cdot triple_i), i = 1, 2, \cdots, k\,, \\ c &= \sum_{i=0}^{k} a_i \cdot \text{cls}(triple_i)\,, \end{aligned} \tag{4}$$

where $\text{cls}_{sequence}$, a_i and c have similar meanings to those in **Tail Only** method.

4.4 Event Prediction

In the three methods in Section 4.3, we all obtain the context representation c of external knowledge sources. And then we fuse c and the sequence representation $\text{cls}_{sequence}$ to get the final score for each candidate choice.

$$\text{score}_i = \text{Linear}([\text{cls}_{sequence}; c]), i = 0, 1, \cdots, m\,, \tag{5}$$

where m is the number of candidate events and the linear function converts the representation from $\mathbb{R}^{2*hidden}$ to $\mathbb{R}^1$, which stands the assigned score to the candidate event. We will perform experiments to verify the effectiveness of three methods.

Finally, we select the candidate event with the highest score as the predicted next event.

$$\text{answer} = \arg\max_i \text{score}_i, i = 1, 2, \cdots, m\,. \tag{6}$$

4.5 Training

Our goal is to minimize the cross-entropy loss between the right answers and the predicted answers given an event chain and a set of choice events. We define the loss function as follows:

$$L(\Theta) = -\frac{1}{N} \sum_{k=1}^{N} \log \frac{\text{e}^{f_{y_i}}}{\sum_{j=1}^{m} \text{e}^{f_{y_j}}}\,, \tag{7}$$

where N is the number of training instances, m is the number of choices in each instance. $f_{y_j} = P(e_{c_j}|e_1, e_2, \cdots, e_n)$ and the i-th choice is the right answer.

5 Experiments

In this section, we introduce the dataset, experiment configuration and compared baselines. We show that our method can achieve state-of-the-art performance on the Multi-Choice Narrative Cloze (MCNC) task. We then dive into the training process to see the effect of external knowledge. Finally, we show a case study to demonstrate how external knowledge can help select the right answer.

5.1 Datasets

The widely used dataset for MCNC task is made public in (Li et al., 2018). This dataset consists of 140,331 training instances, 10,000 development instances and 10,0000 instances for test purpose. There are 5 candidate events for each instance. The goal is to select the candidate event according to the given script. Accuracy is adopted to measure the effectiveness of different models.

In (Lee and Goldwasser, 2019), researchers make a dataset public for testing their model on learning discourse relations, consisting of 28,023 instances. However, the dataset does not consist of training and development dataset. So we do not adopt this dataset in our experiments.

5.2 Baselines

We select popular methods which get good results on the dataset and compare our model with them. The baselines are listed as follows:

- **PMI** (Chambers and Jurafsky, 2008) utilizes PMI to calculate the co-occurrence score between event pairs and sums them together to predict the subsequent event.

- **Bigram** (Jans et al., 2012) utilizes event bi-gram probabilities to predict the subsequent event. They adopt maximum likelihood estimation to learn models.

- **Word2vec** (Le and Mikolov,) represents each component in an event as a word vector and calculates the semantic relations between event pairs.

- **Event-Comp** (Granroth-Wilding and Clark, 2016) proposes a Siamese network to learn the semantic relation among event components of two event and choose the answer with the highest score.

- **PairLSTM** (Wang et al., 2017) first integrates event pair modeling and event chain modeling to predict the subsequent event.

- **SGNN** (Li et al., 2018) constructs an event graph and utilize the graph information to predict the subsequent event. The choice with the highest score will be selected as the predicte answer.

- **SGNN+Int+Senti** (Ding et al., 2019) is based on SGNN and adds external commonsense knowledge like sentiment and intention into event representations.

5.3 Experiment Configuration

We utilize RoBERT-base to represent the input of script and candidate event. We conduct experiments on 4 Nvidia Tesla P100 GPUs. We set batch size to 32 and max length to 64. We adopt Adam optimizer to optimizer our model and the learning rate to 1e-5. We train our model for 10,000 steps. We select the model which gets best result on the development dataset and get its effect on the test dataset. Standard cross-entroy loss is adopted to measure the difference between predicted answer and the right answer.

5.4 Experiment Results and Analysis

The results on the widely used Multi-Choice Narrative Cloze (MCNC) task are shown in Table 1. We can see that our method can achieve state-of-the-art performance and obtain an absolute 2.63% over the best baseline "SGNN+Int+Senti". Although RoBERTa can achieve decent results, by integrating external knowledge our model can achieve much better results. We contribute the results to two main reasons:

Method	Accuarcy(%)
Random	20.00
PMI	30.52
Bigram	29.67
Word2vec	42.23
Event-Comp	49.57
PairLSTM	50.83
SGNN	52.45
SGNN+Int+Senti	56.03*
RoBERTa	56.23
RoBERTa + Tail Only	56.67
RoBERTa + Event Template	58.01
RoBERTa + Representation Fusion	**58.66**

Table 1: Results of script event prediction on the test set. *: SGNN+Int+Senti has many variants and we select the best result they obtain to compare with. Differences between our model and all baseline methods are significant ($p < 0.01$) using t-test.

- The retrieved event knowledge triples contain the necessary event relations which are helpful for script learning problems. Furthermore, our retrieval approach can locate the right event in the knowledge bases and obtain the relevant event knowledge.

- The knowledge representation methods we utilize can represent the extracted event knowledge. And the approach we utilize to integrate external event knowledge can help the model learn the necessary event knowledge.

The second part denotes the comparison among our proposed methods. We can see that three methods to integrate external event knowledge can all bring improvements over the RoBERTa baselines. "RoBERTa + Representation Fusion" method achieves the highest scores. It proves that the event relation plays an important role in predicting the answers and the model can learn the representations for diverse relations.

5.5 Model Training Comparison

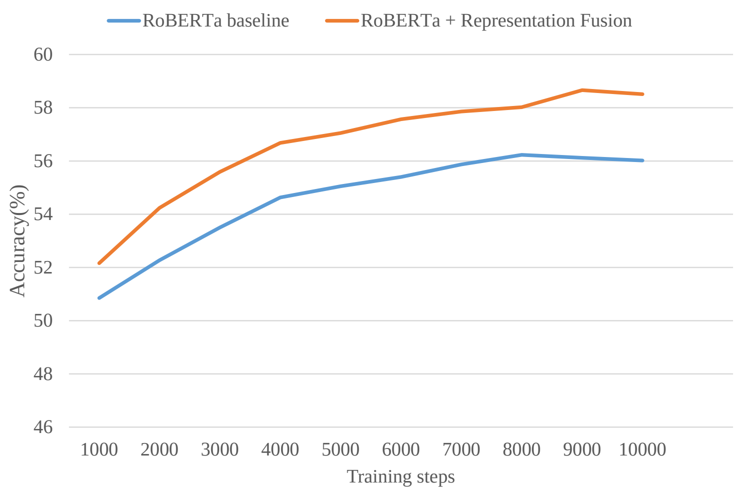

Figure 5: The training process comparison between "RoBERTa" and "RoBERTa + Representation Fusion". We sample the results every 1000 steps.

We have observed that our method achieves better performance compared to other methods. We go one step further to compare the training process of "RoBERTa" and "RoBERTa + Representation Fusion" to see the effect of integrating external knowledge. The result is shown in Figure 5.

From the result curve, we can see that "RoBERTa + Representation Fusion" achieves higher results than "RoBERTa" at every sampled step. When we train the model for about 8000-9000 steps, the model converges and obtains the best result. From the comparison between two models, we can see that by integrating external event knowledge, our model can get better results at the whole training process until convergence.

5.6 Case Study

In this part, we show a case in the development dataset to illustrate how our model can utilize external event knowledge to help predict the right answer.

Script	Candidate Events	External Knowledge	Right Answer
. . . E5: compare basketball E6: buck get basketball E7: whirl basketball bench E8: shout out basketball center	A. look basketball B. weaken basketball C. throw basketball lot youngster D. client deny basketball E. shed basketball	Head: whirl basketball Relation: Precedence Tail: basketball thrown	C

Table 2: Case Study. The script in the dataset contains 8 events. There are 5 candidate events for the script.

From the script we can know that it contains a series of events about "basketball". We read the script and we may infer that a basketball player has got the basketball. And the player "whirl basketball". We can know that the next event may be "he throws the basketball“ or "he passes the basketball to other teammates“. From the candidate event, we can know that "throw basketball" can have a high probability than other events. However, if we do not know the relation between "whirl basketball" and "throw basketball", we will not be able to infer the right answers.

The extracted event knowledge triple "<whirl basketball, precedence, basketball thrown>" denotes that after whirling basketball, the next action will be throwing the basketball. By integrating the knowledge into our model, we can select the right answer will be C. This example shows that external event knowledge can bring benefit to script learning problems when predicting the subsequent event.

6 Conclusion and Future Work

In this work, we propose to integrate external event knowledge into our model to help solve the script learning problem. Our approach consists of two stages: knowledge retrieval and knowledge integration. In the knowledge retrieval stage, we first retrieve relevant events according to the given event. Then we propose to rerank the events according to rule-based scores. In the knowledge integration stage, we propose three methods to integrate external event knowledge and compare the effect of different methods. Experimental results show that our propose method achieve state-of-the-art performance compared to other methods.

In the future work, we will work on retrieve high-quality event knowledge from external event knowledge bases and propose new effective methods to fuse the extracted knowledge to help infer the right answers.

Acknowledgement

We thank the anonymous reviewers for providing valuable suggestions to improve our paper. This research is supported in part by the Beijing Municipal Science and Technology Project under Grant Z191100007119008.

References

Nathanael Chambers and Dan Jurafsky. 2008. Unsupervised learning of narrative event chains. In *Proceedings of ACL-08: HLT*, pages 789–797.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *NAACL-HLT 2019*, pages 4171–4186. Association for Computational Linguistics.

Xiao Ding, Kuo Liao, Ting Liu, Zhongyang Li, and Junwen Duan. 2019. Event representation learning enhanced with external commonsense knowledge. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 4893–4902. Association for Computational Linguistics.

Mark Granroth-Wilding and Stephen Clark. 2016. What happens next? event prediction using a compositional neural network model. In *AAAI*.

Bram Jans, Steven Bethard, Ivan Vulić, and Marie Francine Moens. 2012. Skip n-grams and ranking functions for predicting script events. In *EACL*, pages 336–344.

Quoc V. Le and Tomas Mikolov. Distributed representations of sentences and documents. In *ICML 2014*.

I-Ta Lee and Dan Goldwasser. 2019. Multi-relational script learning for discourse relations. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4214–4226, Florence, Italy, July. Association for Computational Linguistics.

Zhongyang Li, Xiao Ding, and Ting Liu. 2018. Constructing narrative event evolutionary graph for script event prediction. In Jérôme Lang, editor, *IJCAI 2018*, pages 4201–4207. ijcai.org.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Shangwen Lv, Wanhui Qian, Longtao Huang, Jizhong Han, and Songlin Hu. 2019. Sam-net: Integrating event-level and chain-level attentions to predict what happens next. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6802–6809.

Karl Pichotta and Raymond Mooney. 2014. Statistical script learning with multi-argument events. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 220–229.

Karl Pichotta and Raymond J Mooney. 2016. Learning statistical scripts with lstm recurrent neural networks. In *Thirtieth AAAI Conference on Artificial Intelligence*.

Roger C Schank and Robert P Abelson. 2013. *Scripts, plans, goals, and understanding: An inquiry into human knowledge structures*. Psychology Press.

Zhongqing Wang, Yue Zhang, and Ching-Yun Chang. 2017. Integrating order information and event relation for script event prediction. In *EMNLP*. Association for Computational Linguistics, September.

Hongming Zhang, Xin Liu, Haojie Pan, Yangqiu Song, and Cane Wing-Ki Leung. 2020. ASER: A large-scale eventuality knowledge graph. In Yennun Huang, Irwin King, Tie-Yan Liu, and Maarten van Steen, editors, *WWW '20: The Web Conference 2020, Taipei, Taiwan, April 20-24, 2020*, pages 201–211. ACM / IW3C2.