Heterogeneous Graph Neural Networks to Predict What Happen Next

Jianming Zheng, Fei Cai*, Yangxiang Ling, Honghui Chen
Science and Technology on Information Systems Engineering Laboratory
National University of Defense Technology
Changsha, China
{zhengjianming12,caifei,lingyanxiang,chenhonghui}@nudt.edu.cn

Abstract

Given an incomplete event chain, *script learning* aims to predict the missing event, which can support a series of NLP applications. Existing work cannot well represent the heterogeneous relations and capture the discontinuous event segments that are common in the event chain. To address these issues, we introduce a heterogeneous-event (HeterEvent) graph network. In particular, we employ each unique word and individual event as nodes in the graph, and explore three kinds of edges based on realistic relations (e.g., the relations of word-and-word, word-and-event, event-and-event). We also design a message passing process to realize information interactions among homo or heterogeneous nodes. And the discontinuous event segments could be explicitly modeled by finding the specific path between corresponding nodes in the graph. The experimental results on one-step and multi-step inference tasks demonstrate that our ensemble model HeterEvent$_{[W+E]}$ can outperform existing baselines.

1 Introduction

Event chain, also known as *script* (Roger and Robert, 1977), is a structural knowledge format that models stereotypical human activities in a given scenario, e.g., "dining at a restaurant" and "catching a thief" in Fig. 1. Representing such knowledge in a machine-readable way can help machine understand the semantics of natural language and further perform human-like inferences. Besides, event representation can also support a series of downstream applications, such as question answering (Li et al., 2019), discourse understanding (Huang et al., 2019) and information extraction (Liu et al., 2019a; Zheng et al., 2019), text classification (Zheng et al., 2020b), etc.

Existing work on event representation mainly model event chain from three aspects, the intra-event based (Weber et al., 2018; Granroth-Wilding and Clark, 2016), the individual-event based (Li et al., 2018; Wang et al., 2017) and the event-segment (Lv et al., 2019) based models. These methods concentrate on depicting the relations among homogeneous modeling objectives, e.g., the inter-event relations. However, the relations among heterogeneous ones, e.g., the subordinated relations between word and event, which are also critical to the event chain, have not yet been taken into account in previous work. Besides, (Lv et al., 2019) found the *event segments*, a set of individual events related to each other, were helpful to predict the missing event and event segments could be continuous or discontinuous (See Fig. 1). Although the self-attention mechanism can implicitly represent such discontinuous event segments by greedily making connections among all events (Lv et al., 2019), it inevitably introduces noises.

In this paper, we attempt to deal with these issues by proposing a heterogeneous-event (HeterEvent) graph network . Specifically, we define two different types of nodes in the HeterEvent graph, including *word* and *event* nodes, which respectively represent unique words and individual events in the event chain. Then we construct three types of edges, namely *word-word* edges denoting the co-occurrence relations within word nodes, *word-event* edges denoting the subordinate relations between word and event nodes, and *event-event* edges denoting the order relations within event nodes. We also design a message

* Corresponding Author

Proceedings of the 28th International Conference on Computational Linguistics, pages 328–338
Barcelona, Spain (Online), December 8-13, 2020

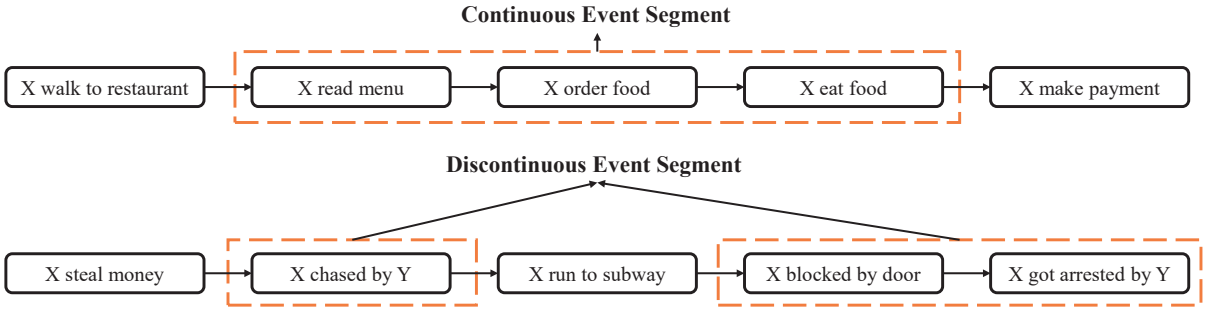

Figure 1: Examples of Continuous and Discontinuous Event Segments

passing process to realize information interactions among homo or heterogeneous nodes. Furthermore, the HeterEvent graph can explicitly represent discontinuous event segments by finding a path between their corresponding nodes in the graph. We evaluate our proposal on one-step (Granroth-Wilding and Clark, 2016) and multi-step (Lee and Goldwasser, 2018) inference tasks, and the experimental results prove that our proposals present a stronger inference ability than existing baselines in event prediction.

Our research contributions in this paper are in three folds:

1. To the best of our knowledge, we are the first to construct a heterogeneous graph network to model the event chain.
2. Our model outperforms the existing baselines on one-step and multi-step inference tasks.
3. Our proposed HeterEvent is a expandable framework that can be easily adapted to other granularities of information nodes, e.g., subwords or event scenario, which both are a part of the event chain.

2 Related Work

Event chain models human understandings about the relevant causal relationships among events. An event chain can be used to infer how events will unfold in a given scenario (Roger and Robert, 1977). Restricted to the manual acquisition, early work on event chain shows a slow progression until narrative event chains introduced by (Chambers and Jurafsky, 2008). (Chambers and Jurafsky, 2008) assumed that although a narrative script had several participants, there was a central actor (i.e., protagonist) who characterized a narrative chain. In this assumption, probabilistic co-occurrence-based models combined with dependency parser can realize the automatic extraction of narrative event chains from raw text. They also casted narrative events as the format $< predicate, dependency\_type >$, where the *predicate* was a verb lemma and the *dependency_type* denoted a grammatical dependency relation between the *predicate* and the protagonist, e.g., 'subj', 'obj' or 'iobj'. Besides, (Pichotta and Mooney, 2014) explored a richer representation over multi-argument event format.

From the perspective of modeling objectives, existing event representation works can be classified into three main types, i.e., the *intra-event* based, the *individual-event* based and the *event-segment* based models. Firstly, intra-event based methods concentrate on the multiplicative interaction among intra-event elements. For instance, (Granroth-Wilding and Clark, 2016) simply concatenated predicate and argument embeddings and fed them into a neural network to get the event representation. While (Weber et al., 2018) used the tensor-network-based model to capture more subtle semantic interactions. Secondly, individual-event based methods mainly investigate the complex and diverse relations between two individual events. (Wang et al., 2017) utilized the LSTM hidden states to integrate the chain order information into event model. (Li et al., 2018) extended the narrative event chains into the narrative event evolutionary graph to model the dense connections among events. While (Lee and Goldwasser, 2019) broadened the single relation (time-order relation) into the diverse ones based on the discourse relations from PDTB (Prasad et al., 2008). Thirdly, event-segment based methods focus on a set of semantic-related events. (Lv et al., 2019) developed self-attention mechanism to implicitly model relations in event segments. (Zheng et al., 2020a) proposed a unified fine-tuning framework to integrate the training losses from different layers. Besides, by jointly training the event representation model and external knowledge, some work intended to mine the potential connections between narrative event chains and external knowledge. (Ding et al., 2019) introduced ATOMIC (Sap et al., 2019) to obtain the sentiment

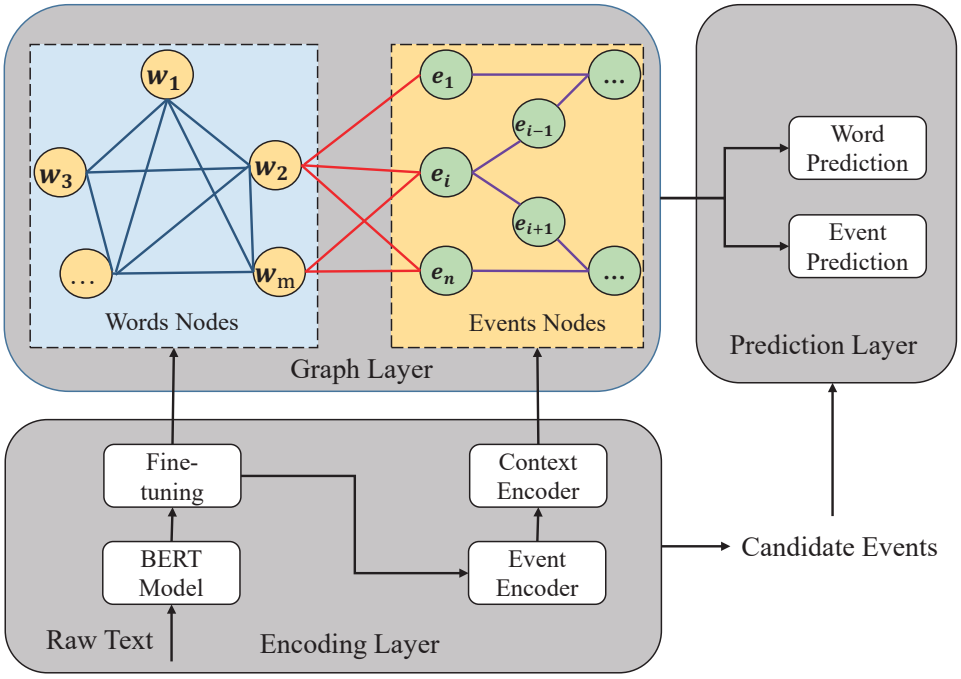

Figure 2: The overall architecture of HeterEvent. In the graph layer, yellow and green circles denote word nodes and event nodes, respectively; blue, red and purple lines denote word-word, word-event and event-event edges, respectively.

and intent information of event.

Different from the above methods, our work attempts to synthetically represent multi-granularity information and discontinuous event segments contained in a event chain by a heterogeneous graph network, which can provide strong inferring abilities on the event prediction It also worths noting that the HeterEvent graph is a scalable framework that can be easily adjusted to fusion other grained information, e.g., subwords or event scenario.

3 Methodology

In this paper, we aim at learning the event representation to predict the missing event. The research problem of event prediction in a narrative event chain is defined as follows. Given an incomplete event chain $\{e_1, e_2, \cdots, e_n\}$ and a set of candidate events $\{e_{c_1}, e_{c_2}, \cdots, e_{c_m}\}$, our goal is to choose the correct one from candidate events for the missing event in the event chain.

As Fig. 2 shows, the overall architecture of HeterEvent can be divided into the following three components: an encoding layer, a graph layer and a prediction layer. (1) *Encoding layer* aims to transform words and events into a distributed representation. (2) *Graph Layer* first performs the heterogeneous graph construction, where each individual word and event are defined as nodes, three types of relations including word-word, word-event and event-event, are extracted as edges. Then the message passing layer is employed to realized information interactions among homo or heterogeneous nodes. (3) *Prediction Layer* calculates probabilities of candidate events as the missing event conditioned on the representation learned from the graph layer.

3.1 Encoding Layer

3.1.1 BERT and Fine-tuning

To overcome the inconsistence of pre-trained corpus, where the BERT model was pre-trained on BooksCorpus (Zhu et al., 2015) and English Wikipedia (Devlin et al., 2019) while narrative event chains on the Gigaword corpus (Graff et al., 2003), we employ a fine-tuning method to minimize such inconsistency. Similar to the masked language model (Devlin et al., 2019; Liu et al., 2019b), we randomly mask some words in a text sequence with $[mask]$ tokens, and feed them into the BERT model to predict

those masked words. Different from previous methods (Granroth-Wilding and Clark, 2016; Weber et al., 2018; Lv et al., 2019) that directly apply GloVe (Pennington et al., 2014) as the word representation, such fine-tuning narrows the semantic distribution gap when transferring to different corpus.

3.1.2 Event and Context Encoder

Since each event consists of three types of intra-event elements, i.e., subject, predicate and object, so we first employ a max pooling and an average pooling on words respectively, and then concatenate them to get the representation for three intra-event elements (the subject, predicate and object representation are denoted as $s(e), p(e), o(e) \in \mathbb{R}^{2d}$, respectively). Specially, the subject representation $s(e)$ can be defined as follows:

$$s(e) = [max([w_{s,1}; w_{s,2}; \cdots; w_{s,n_s}]); ave([w_{s,1}; w_{s,2}; \cdots; w_{s,n_s}])], \tag{1}$$

where $w_{s,1}; w_{s,2}; \cdots; w_{s,n_s} \in \mathbb{R}^d$ are the representation for words in the subject, $max(\cdot)$, $ave(\cdot)$, and $[;]$ denote the max-pooling , average-pooling and concatenating operations, respectively. The same strategy is also applied to obtain the representation for the predicate ($p(e)$) and the object ($o(e)$).

Following (Weber et al., 2018), we adopt a tensor-based model (Socher et al., 2013) to model subtle semantic interactions among intra-event elements. Given a 3-dimension tensor based network $T(,)$ with two inputs a and b where $T \in \mathbb{R}^{d \times 2d \times 2d}, a, b \in \mathbb{R}^{2d}$, we can get the computation result as $T(a, b) = \sum_{j,k} T_{i,j,k} a_j b_k$. Hence, the representation $e(e)$ for each individual event can be formulated as follows:

$$e(e) = W_s T(s(e), p(e)) + W_o T(o(e), p(e)) \tag{2}$$

where $W_s, W_o \in \mathbb{R}^{d \times d}$ are the trade-off matrices for the subject role and the object role, respectively.

Furthermore, we use a Bi-GRU on top of the event encoder to model temporal interactions between events, i.e., forward and backward order information (Wang et al., 2017). Hence, we can obtain a sequence of hidden state representation $\{h_1, h_2, \cdots, h_n\}$ by recurrently feeding the event representation $\{e(e_1), e(e_2), \cdots, e(e_n)\}$ as inputs to the Bi-GRU, i.e.,

$$\begin{cases} \overleftarrow{h_i} = \overleftarrow{GRU}(e(e_i), \overleftarrow{h_{i-1}}) \\ \overrightarrow{h_i} = \overrightarrow{GRU}(e(e_i), \overrightarrow{h_{i-1}}) \end{cases}, \tag{3}$$

where $h_i = [\overleftarrow{h_i}; \overrightarrow{h_i}]$, h_0 and other parameters in Bi-GRU are randomly initialized.

3.2 Graph Layer

3.2.1 HeterEvent Graph Construction and Initialization

Let a HeterEvent graph be denoted as $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, where $\mathcal{V}$ stands for node and $\mathcal{E}$ represents edges between nodes. In particular, we treat each individual word and event as nodes (i.e., $\mathcal{V} = \mathcal{V}_w \cup \mathcal{V}_e$), and define three types of undirected edges between pair of nodes to model various structural information in the HeterEvent graph (i.e., $\mathcal{E} = \{\mathcal{E}_{w-w} \cup \mathcal{E}_{w-e} \cup \mathcal{E}_{e-e}\}$). Here, $\mathcal{V}_w = \{w_1, w_2, \cdots, w_m\}$ denotes m unique words in an event chain and $\mathcal{V}_e = \{e_1, e_2, \cdots, e_n\}$ corresponds to n individual events in the event chain. For edge connection, the definitions of there types of edge are as follows:

- $\mathcal{E}_{w-w}$: an edge between two word nodes if two words co-occur in the same event;
- $\mathcal{E}_{w-e}$: an edge between a word node and an event node if the word appears in the event;
- $\mathcal{E}_{e-e}$: an edge between two event nodes if two events are adjacent in the event chain.

In the graph layer of Fig. 2, we illustrate a toy example of our proposed HeterEvent graph, where yellow and green circles stand for word nodes $\mathcal{V}_w$ and event nodes $\mathcal{V}_e$, respectively; while blue, red and purple lines denote $\mathcal{E}_{w-w}$, $\mathcal{E}_{w-e}$ and $\mathcal{E}_{e-e}$ edges.

As for the initialization of HeterEvent graph, we adopt the word representation from fine-tuned BERT model as the representation of word nodes, and the hidden state representation of event from context

encoder as the representation of event nodes. For edge, we treat these three types of edges as the same level to propagate information over edges.

Generally speaking, our HeterEvent graph consists of two types of nodes: word and event nodes. Either homo or heterogeneous nodes could be connected by three types of relations. And physically-divided but semantics-related nodes (e.g., the discontinuous event segment) can establish relationships by finding a path in the graph. Furthermore, we can readily add more granularities of information nodes (e.g., subwords or event segments) into the HeterEvent graph.

3.2.2 Message Passing

Now we have an initialized HeterEvent graph, the next step is to make information of each node pass to each other over edges. Existing information passing methods (e.g., graph convolutional networks (Kipf and Welling, 2017), graph attention networks (Velickovic et al., 2018)) in graph neural networks mostly based on a neighborhood aggregation strategy, in which the update of the node representation depends on the information aggregation of neighborhood nodes.

Formally, given a node i and its neighborhood nodes set $\mathcal{N}_i$, the output of neighborhood aggregation for node i in layer k can be formulated as follows:

$$z_i^k = \sigma(\sum_{j \in \mathcal{N}_i} \alpha_{ij} h_j^k), \tag{4}$$

where σ denotes the sigmoid function, h_j^k is the node representation of node j in layer k. Similar to (Velickovic et al., 2018), α_{ij} is the attention weight between h_i^k and h_j^k, which is defined as follows:

$$\alpha_{ij} = \frac{exp(LeakyReLU(W_a[W_s h_i^k; W_s h_j^k]))}{\sum_{l \in \mathcal{N}_i} exp(LeakReLU(W_a[W_s h_i^k; W_s h_l^k]))}, \tag{5}$$

where $LeakyReLU(\cdot)$ is a nonlinear function, W_a, W_s are trainable weight matrices, and $[;]$ is a concatenation operation.

For graph neural networks, when the number of layers is too large, i.e., neighborhood aggregation is conducted too many times, they easily suffer from the over-smoothing problem (Kipf and Welling, 2017). Hence, we add a residual connection (He et al., 2016) to deal with this problem:

$$u_i^k = z_i^k + h_i^k \tag{6}$$

Besides, we also introduce an information gate g_i^k to control the update process of the node representation h_i^k (Tu et al., 2019). Therefore, the updated representation of node i, i.e., the representation of node i in layer $k+1$, can be represented as:

$$\begin{cases} h_i^{k+1} = g_i^k \odot tanh(u_i^k) + (1 - g_i^k) \odot h_i^k \\ g_i^k = \sigma(W_g[u_i^k; h_i^k]) \end{cases}, \tag{7}$$

where $\odot$ means the element-wise multiplication, σ is the sigmoid function, and W_g is a trainable weight matrix. Hence, following such information passing strategy, adding one information passing layer can realize the information aggregation of one-hop neighborhood nodes, i.e., continuous nodes. That is, one more information passing layer is equivalent to the information aggregation of one-more-hop neighborhood nodes, which help information pass to discontinuous nodes.

3.3 Prediction Layer

After the graph layer, we have got the node representation for event, i.e., $\{h_{e_1}, h_{e_2}, \cdots, h_{e_n}\}$. For $h_{e_i} (i = 1, 2, \cdots, n)$, the node representation for its subordinate words is $\{w_1^{e_i}, w_2^{e_i}, \cdots, w_{n_{e_i}}^{e_i}\}$, where n_{e_i} is the number of words in e_i. In the training phase, we consider to train our HeterEvent model from two aspects: the word level and the event level.

In the word level, the node representation of words should have the ability to predict its neighborhood word nodes, i.e., the training of $\mathcal{E}_{e-e}$. On the other hand, the words nodes should also be inferred by the node representation of their source events, i.e., the training of $\mathcal{E}_{w-e}$. Therefore, the loss function in the word level can be formulated as follows:

$$\mathcal{L}_w = \frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{n_{e_i}} -log(P(w_j^{e_i}|w_1^{e_i})P(w_j^{e_i}|h_{e_i})) + \lambda\mathcal{L}_w(\theta_w), \tag{8}$$

where $P(w_j^{e_i}|h_{e_i}), P(w_j^{e_i}|h_{e_i})$ are computed via a softmax layer, λ is a trade-off parameter, and $\mathcal{L}_w(\theta_w)$ is $l2$ regularization on all parameters θ_w. In the event level, the node representation of event h_{e_i} should have the ability to predict the previous and the next event, i.e., the training of $\mathcal{E}_{s-s}$. Similarly, the loss function in the event level can be formulated as follows:

$$\mathcal{L}_e = \frac{1}{n}\sum_{i=1}^{n} -log(P(h_{e_{i-1}}|h_{e_i})P(h_{e_{i+1}}|h_{e_i})) + \lambda\mathcal{L}_e(\theta_e), \tag{9}$$

where $P(h_{e_{i-1}}|h_{e_i}), P(h_{e_{i+1}}|h_{e_i})$ are also computed via a softmax layer, especially $P(h_{e_0}|h_{e_1}) = 1$, and $\mathcal{L}_w(\theta_e)$ is $l2$ regularization on all parameters θ_e. In addition, we also introduce a combined loss $\mathcal{L}_{w+e}$ by a simple addition, i.e., $\mathcal{L}_{w+e} = \mathcal{L}_w + \mathcal{L}_e$.

In the testing phase, the whole event chain follows the encoder layer and the graph layer to construct the HeterEvent graph. While candidate events only pass the encoder layer to obtain corresponding representation, which are combined with the constructed graph to select the most probable one based on a softmax layer.

4 Experiments

4.1 Evaluation Tasks

We evaluate our proposed models on two types of inference tasks: one-step and multi-step inference tasks.

One-step Inference Task aims to predict a missing event given its context. Based on this, (Granroth-Wilding and Clark, 2016) proposed the multiple-choice narrative cloze (MCNC) dataset.

Multi-step Inference Task extended from the one-step inference task, evaluates the model's ability to make longer inferences, instead of just predicting one event. (Lee and Goldwasser, 2018) proposed three selection strategies to construct event chains, e.g., *Viterbi*, *Base* and *Sky*. *Viterbi* considers the integrity of event chain and finds the most probable event chain; *Base* greedily picks the best transition and then moves to the next time stamp; *Sky* breaks down a sequence of prediction into individual decisions which applies the golden states of all contextual events. Hence, theses three selection strategies can build four versions of multi-inference datasets, i.e., MCNS-V, MCNE-V, Base and Sky, where MCNS-V and MCNE-V are both constructed by *Viterbi* and have a start event, except MCNE-V has an additional end event; while Base and Sky are constructed by *Base* and *Sky* algorithms, respectively.

Furthermore, the above inference datasets are all in a multiple-choice setting, i.e., the event representation model should choose a positive event from one golden choice and four corrupted choices for each-step inference.

4.2 Model Summary

According to the model taxonomy in Section 2, we first select some recent models as baselines, which are shown as follows. **Event-comp:** an intra-event based method that consists of intra-event elements based on a fully connected network (Granroth-Wilding and Clark, 2016). **Role-factor:** an intra-event based method that models multiplicative interactions among intra-event elements based on a tensor network (Weber et al., 2018). **EventTransE:** an individual-event based method that explores the inter-event relation based on the discourse relations (Lee and Goldwasser, 2019). **SAM-Net:** an event-segment based method that explores the event-segment relations (Lv et al., 2019). **FEEL:** an external-knowledge based method that introduces the sentiment and animacy information (Lee and Goldwasser, 2018). **IntSent:** an

Model	One-step Inference	Multi-step Inference			
	MCNC	MCNS-V	Base	Sky	MCNE-V
Event-comp	46.3	29.9	27.8	38.4	32.5
Role-factor	48.8	28.6	28.3	39.6	32.5
EventTransE	63.7	59.5	51.2	64.5	60.9
SAM-Net	54.3	46.2	43.2	50.4	49.2
FEEL	51.6	41.6	38.5	46.0	44.8
IntSent	56.4	44.7	42.2	49.6	48.5
HeterEvent$_{[W]}$	62.6	58.7	48.9	63.2	59.1
HeterEvent$_{[E]}$	63.5	59.8	50.7	65.4	60.8
HeterEvent$_{[W+E]}$	**64.4**$^{\blacktriangle}$	**60.3**$^{\triangle}$	**51.3**$^{\triangle}$	**65.7**$^{\blacktriangle}$	**61.7**$^{\triangle}$

Table 1: The inference performance in terms of Accuracy (%). The results produced by the best baseline and the best performer in each column are underlined and boldfaced, respectively. Statistical significance of pairwise differences of HeterEvent$_{[W+E]}$ vs. the best baseline is determined by a t-test ($^{\blacktriangle}$ for α = .01, or $^{\triangle}$ for α = .05).

external-knowledge based method that introduces the intent and sentiment information (Ding et al., 2019).

Next, we list the models proposed in this paper for comparison. **HeterEvent$_{[L]}$** : a heterogeneous graph based model with a specific loss $[L]$, e.g., the word-level loss (HeterEvent$_{[W]}$), the event-level loss (HeterEvent$_{[E]}$) and the combined loss (HeterEvent$_{[W+E]}$).

4.3 Model Configuration

Following (Granroth-Wilding and Clark, 2016; Lee and Goldwasser, 2018; Lee and Goldwasser, 2019), we choose the New York Times portion of the Gigaword corpus [1] as the raw-text corpus. In addition, we use the Stanford CoreNLP (Manning et al., 2014) to extract the dependency parses and coreference chains. Based on the coreference chains, we creat the event chains in the form of $(pred, subj, obj)$. For the extraction of intra-event words, we keep the complete mention spans rather than only headwords. The detailed extraction process can refer to (Lee and Goldwasser, 2019). Finally, we select 1.4M event chains as the training set, 10K event chains as the development set and 10K event chains as the test set (#MCNC, #MCNS-V, #MCNE-V, #Base, #Sky in the test set are 2k, 2k, 2k, 2k and 2k, respectively.).

During training, we set the batch size to 128 and regularization weight to 10^{-5}. We adopt the Adam Optimizer (Kingma and Ba, 2015) with exponential-descent learning rate to optimizer the loss. We also used gradient clipping with a threshold of 10 to stabilize GRU training (Pascanu et al., 2013). As for word embeddings, we adopt the pre-trained bert-base-uncased version to initialize the model and refer readers to (Devlin et al., 2019) for details. Other weighted or trade-off matrices are initialized with Xavier Initialization (Glorot and Bengio, 2010). Specially, we also employ the same parameters on all models for each inference datasets.

4.4 Overall Evaluation Results

We examine the inference ability of our proposal as well as the baselines for the one-step inference task (i.e., MCNC) and the multi-step inference task (i.e., MCNS-V, Base, Sky and MCNE-V) , respectively. For comparison, we present the experimental results in Table 1.

Firstly, we zoom in the comparison among baselines. Among models without a graph-based structure, the best baseline for evaluating the inference performance is EventTransE (Lee and Goldwasser, 2019). Compared with other methods, EventTransE gains advantages over the introduction of the PDTB corpus (Prasad et al., 2008) that can provide additional prior knowledge to enrich the inter-event relations.

[1] https://catalog.ldc.upenn.edu/LDC2003T05

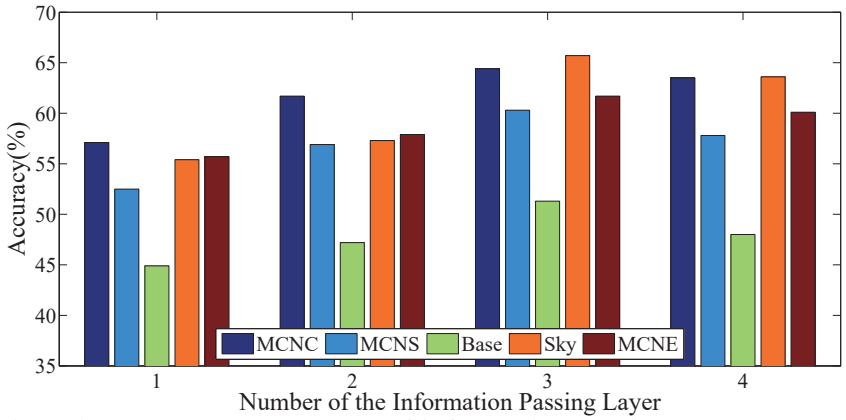

Figure 3: Relationships between the number of information passing layer and the inference performance for HeterEvent$_{[W+E]}$.

Next, we compare our proposals with baselines. Clearly, our ensemble model with the combined loss, i.e., HeterEvent$_{[W+E]}$, can outperform all discussed models in the inference performances. Its advantages against baselines indicate that the connection between the given event chain and the missing event can be mined and captured by not only homo or heterogeneous relations but also explicit multi-hop relations in a graph-based structure. In addition, significant improvements against the best baseline are observed for HeterEvent$_{[W+E]}$ at the α = .05 level on MCNC and Sky, while at the α = .01 level on MCNS-V, Base and MCNE-V. Such differences may be explained by the fact that longer inference steps increase the inference difficulty, thus making the multi-step inference task more challenging. Furthermore, two individual losses (i.e., HeterEvent$_{[W]}$ and HeterEvent$_{[E]}$) both fail in the comparison with the combined one (HeterEvent$_{[W+E]}$), which may be attributed to the fact that two individual losses concentrate on different part of event chain, and the combined loss can help integrate these two advantages.

4.5 Analysis of the Message Passing Layer

As shown in Sec. 3.2.2, one information passing layer is equivalent to the information aggregation of one-hop neighborhood nodes. Intuitively, the more information passing layer, the deeper the information interactions among nodes in the heterogeneous graph. However, adding too many information passing layer will cause graph-based methods to suffer from the over-smoothing problem (Kipf and Welling, 2017). Hence, it is important to explore the influence of the number of the information passing layers on the inference performance.

In Fig. 3, we present results of HeterEvent$_{[W+E]}$ in various number of the information passing layer (denoted as l, $l = 1, 2, 3, 4$) for different inference datasets. We can clearly observe that as the number of layer increases, the performance of HeterEvent$_{[W+E]}$ in any dataset always increases first to reach the best performance and then drops. In particular, HeterEvent$_{[W+E]}$ always achieve the best performance when the layer number increases to 3. These consistent patterns may be attributed to the size of graph, most of which has less than 50 nodes. 3 information passing layers can realize the aggregation of 3-hop neighborhood nodes for each node, which can achieve a minimum coverage of nodes in the graph. Once more than 3 layers, the over-smoothing problem will be amplified.

4.6 Analysis of the Average Nodes Degree

For the whole graph, the average degree of all nodes measures the overall connection level of the graph. On the other hand, it can also reflect the closeness between the given event chain and the missing event. So it is meaningful to explore the impact of the average degree of all nodes on the inference performance.

We first calculate the average node degree for each example based on respective constructed graph. For simplicity, we don't distinguish the degree for different types of nodes. Based on the distribution of the average node degree, we roughly divide each test example into six intervals (x-axis in Fig. 4), i.e., $(0, 4)$, $[4, 4.5)$, $[4.5, 5)$, $[5, 5.5)$, $[5.5, 6)$, and $[6, \infty)$. We group the performance of HeterEvent$_{[W+E]}$ in five inference datasets based on the set intervals and present them in the Fig. 4. From Fig. 4, we can

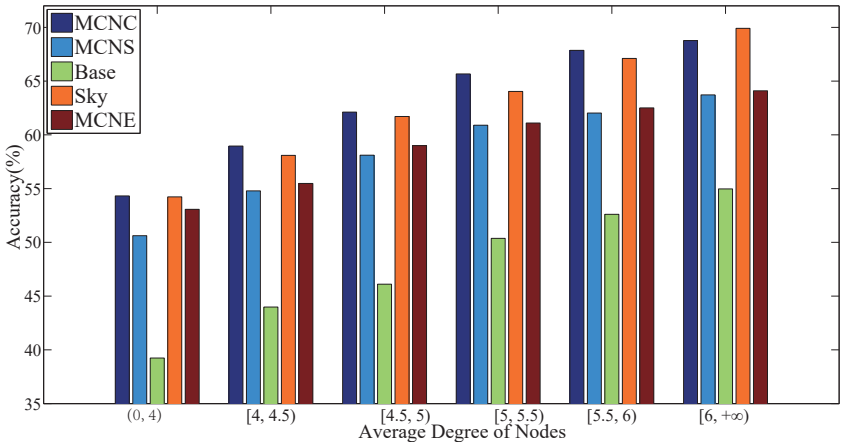

Figure 4: Relationships between the average node degree and the inference performance for HeterEvent$_{[W+E]}$.

Model	One-step Inference	Multi-step Inference			
	MCNC	MCNS-V	Base	Sky	MCNE-V
HeterEvent$_{[W+E]}$	64.4	60.3	51.3	65.7	61.7
− BERT	59.6	55.0	45.9	62.8	56.6
− Context Encoder	63.5	59.3	50.2	64.7	61.0
− Residual Connection	62.7	59.5	50.1	63.6	60.7
− Word Nodes	58.1	53.5	41.6	59.4	55.7
− Event Nodes	50.8↓	35.3↓	36.1↓	46.0↓	40.2↓

Table 2: Ablation studies of HeterEvent$_{[W+E]}$ on inference datasets. The biggest drop in each column is appended ↓.

clearly observe that in any dataset, the performance of HeterEvent$_{[W+E]}$ gets a stable boost in terms of accuracy with the growth of the average node degree. This uniform mode proves that the higher average node degree reflects the higher overall connection level of the graph, which is easier for the HeterEvent graph to make inferences.

4.7 Ablation Studies

In order to better understand the contribution of different modules to the inference performance, we conduct ablation studies using our proposed HeterEvent$_{[W+E]}$ on five inference datasets. In the ablation studies, we remove or replace some specific layers or modules and explore their influence on our proposed models, which is denoted as the notation '-'. For example, '−BERT' means replacing the BERT layer in HeterEvent$_{[W+E]}$ with the GloVe embedding matrix (Pennington et al., 2014); '−Context Encoder' and '−Residual Connection' denote directly removing this component; '−Word Nodes' and '−Event Nodes' respectively remove word nodes and event nodes in the graph, including relations connected to the removed nodes. We present their ablation results in Table 2.

In detail, '−BERT' causes a marginal decline of performance on inference datasets, which indicates that BERT is a better embedding method than GloVe. Obvious performance declines in '−Context Encoder' verify that modeling temporal relations can help predict the missing event. While the degeneration in '−Residual Connection' demonstrates that the residual connection can mitigate the effect of the over-smoothing problem. In the heterogeneous graph, '− Word Nodes ' and '− Event Nodes ' both cause sharp performance declines for inference datasets, which prove that these two types of node are both indispensable to make event inferences. Besides, the comparison between'− Event Nodes ' and '− Word Nodes ' implies that event nodes have a greater impact on inferences than word nodes.

5 Conclusion and Future Work

In this paper, we introduce a novel heterogeneous-event graph network (HeterEvent) for the event representation to predict the missing event. Based on the characteristics of event chain, we explore two types of nodes (i.e., word and event nodes) and three kinds of edges (i.e., the relations of word-and-word, word-and-event, event-and-event) to construct a heterogeneous graph. Experimental results on five inference datasets demonstrate that our graph-based model can effectively encode homo and heterogeneous relations as well as multi-hop connections, which help HeterEvent$_{[W+E]}$ to achieve the best performance compared to all discussed models.

As to future work, on the one hand, we plan to investigate how to incorporate more granularities of nodes into the heterogeneous graph, e.g., subwords, event segments or event scenario. On the other hand, we plan to further extend and refine the type of edges, since multi types of inter-event relations have been proven effective in EventTransE (Lee and Goldwasser, 2019).

Acknowledgements

This work was partially supported by the National Natural Science Foundation of China under No. 61702526, the Defense Industrial Technology Development Program under No. JCKY2017204B064, the Postgraduate Scientific Research Innovation Project of Hunan Province under No. CX20190034. All content represents the opinion of the authors, which is not necessarily shared or endorsed by their respective employers and/or sponsors.

References

Nathanael Chambers and Daniel Jurafsky. 2008. Unsupervised learning of narrative event chains. In *ACL*, pages 789–797. ACL.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, pages 4171–4186. ACL.

Xiao Ding, Kuo Liao, Ting Liu, Zhongyang Li, and Junwen Duan. 2019. Event representation learning enhanced with external commonsense knowledge. In *EMNLP-IJCNLP*, pages 4893–4902. ACL.

Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *AISTATS*, volume 9 of *JMLR Proceedings*, pages 249–256. JMLR.org.

David Graff, Kong Junbo, and Chenand Kazuaki Maeda Ke. 2003. English gigaword. *Linguistic Data Consortium, Philadelphia*, 4(2):34.

Mark Granroth-Wilding and Stephen Clark. 2016. What happens next? event prediction using a compositional neural network model. In *AAAI*, pages 2727–2733. AAAI Press.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *CVPR*, pages 770–778. IEEE Computer Society.

Rongtao Huang, Bowei Zou, Hongling Wang, Peifeng Li, and Guodong Zhou. 2019. Event factuality detection in discourse. In *NLPCC*, volume 11839, pages 404–414. Springer.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR*. OpenReview.net.

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *ICLR*. OpenReview.net.

I-Ta Lee and Dan Goldwasser. 2018. FEEL: featured event embedding learning. In *AAAI*, pages 4840–4847. AAAI Press.

I-Ta Lee and Dan Goldwasser. 2019. Multi-relational script learning for discourse relations. In *ACL*, pages 4214–4226. ACL.

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

Feng-Lin Li, Kehan Chen, Yan Wan, Weijia Chen, Qi Huang, and Yikun Guo. 2019. Using event graph to improve question answering in e-commerce customer service. In *ISWC*, volume 2456, pages 327–328. CEUR-WS.org.

Xiao Liu, Heyan Huang, and Yue Zhang. 2019a. Open domain event extraction using neural latent variable models. In *ACL*, pages 2860–2871. ACL.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/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 *AAAI*, pages 6802–6809. AAAI Press.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Rose Finkel, Steven Bethard, and David McClosky. 2014. The stanford corenlp natural language processing toolkit. In *ACL*, pages 55–60. ACL.

Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. 2013. On the difficulty of training recurrent neural networks. In *ICML*, volume 28, pages 1310–1318. JMLR.org.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*, pages 1532–1543. ACL.

Karl Pichotta and Raymond J. Mooney. 2014. Statistical script learning with multi-argument events. In *EACL*, pages 220–229. ACL.

Rashmi Prasad, Nikhil Dinesh, Alan Lee, Eleni Miltsakaki, Livio Robaldo, Aravind K. Joshi, and Bonnie L. Webber. 2008. The penn discourse treebank 2.0. In *LREC*. European Language Resources Association.

Schank Roger, C and Abelson Robert, P. 1977. Scripts, plans, goals, and understanding: An inquiry into human knowledge structures. *Lawrence Erlbaum Associates*.

Maarten Sap, Ronan Le Bras, Emily Allaway, Chandra Bhagavatula, Nicholas Lourie, Hannah Rashkin, Brendan Roof, Noah A. Smith, and Yejin Choi. 2019. ATOMIC: an atlas of machine commonsense for if-then reasoning. In *AAAI*, pages 3027–3035. AAAI Press.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *ACL*, pages 1631–1642. ACL.

Ming Tu, Guangtao Wang, Jing Huang, Yun Tang, Xiaodong He, and Bowen Zhou. 2019. Multi-hop reading comprehension across multiple documents by reasoning over heterogeneous graphs. In *ACL*, pages 2704–2713. Association for Computational Linguistics.

Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph attention networks. In *ICLR*. OpenReview.net.

Zhongqing Wang, Yue Zhang, and Ching-Yun Chang. 2017. Integrating order information and event relation for script event prediction. In *EMNLP*, pages 57–67. ACL.

Noah Weber, Niranjan Balasubramanian, and Nathanael Chambers. 2018. Event representations with tensor-based compositions. In *AAAI*, pages 4946–4953. AAAI Press.

Jianming Zheng, Fei Cai, Wanyu Chen, Chong Feng, and Honghui Chen. 2019. Hierarchical neural representation for document classification. *Cognitive Computation*, 11(2):317–327.

Jianming Zheng, Fei Cai, and Honghui Chen. 2020a. Incorporating scenario knowledge into A unified fine-tuning architecture for event representation. In *SIGIR*, pages 249–258. ACM.

Jianming Zheng, Fei Cai, Honghui Chen, and Maarten de Rijke. 2020b. Pre-train, interact, fine-tune: a novel interaction representation for text classification. *Information Processing & Management*, page 102215.

Yukun Zhu, Ryan Kiros, Richard S. Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *ICCV*, pages 19–27. IEEE Computer Society.