Neurocomputing 407 (2020) 376–384

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

A neural model for joint event detection and prediction

Linmei Hu [a], Shuqi Yu [a], Bin Wu [a,*], Chao Shao [b], Xiaoli Li [c]

[a] School of Computer Science, Beijing University of Posts and Telecommunications, Beijing, China
[b] Tsinghua University, Beijing, China
[c] Institute for Infocomm Research, Singapore

ARTICLE INFO

Article history:
Received 2 July 2019
Revised 7 April 2020
Accepted 18 May 2020
Available online 26 May 2020
Communicated by Jing Jiang

2010 MSC:
00-01
99-00

Keywords:
Event prediction
Joint event detection and prediction
Hierarchical attention

ABSTRACT

Event prediction aims to predict the future possible event given a sequence of previously happened events. Event prediction is important since it can benefit the government, agencies and companies for avoiding damages by taking proactive actions. A further related task is event detection, which is to classify each event to predefined types, helping users quickly find relevant information. Event prediction is related to event detection, since salient information of events is universal between the tasks. In this paper, we propose a novel neural model for joint event detection and prediction, which classifies the events to predefined types as well as predicts the next probable event by generating a sequence of words describing it. In addition, we propose a hierarchical attention mechanism to enable the model to capture important information at both word level and event level for next event prediction. Empirical experiments on a real-world dataset reveal that our joint model with hierarchical attention achieves substantial improvements on event prediction, advancing state-of-the-art models. With joint learning, our model also improves the performance on event detection.

© 2020 Elsevier B.V. All rights reserved.

1. Introduction

An event is a specific thing that happens at a particular time and place [1]. Typically, an event could trigger a series of following events. Over years, a massive amount of news series containing sequences of events have been accumulated. We are supposed to learn common sequential patterns from the large amount of news series, and empowered to predict the future events given a sequence of existing events.

Predicting future events given a sequence of previously happened events is quite meaningful for the governments, companies and individuals to take proactive measures by providing predictive information. Take a news series describing "11.13 Paris Terror Attack" for example, given the event sequences where each event is represented by a news title, "Explosion occurs outside a sports stadium in France", "A mass shooting and hostage-taking occurred", "President declares a state of emergency after the attacks unfold", "ISIS claims responsibility for the attacks", our model generates "France", "Paris", "shooting", "sever","emergenc y", "chaos", word by word. It is consistent with the true next event "Terrorist attack in France caused injuries and chaos". With the prediction of chaos, the government is expected to take proactive or preventive measures to avoid casualties and damages.

Existing works dedicated to future event prediction can be discussed in two lines. Most of the works focus on predicting target (known) events [2,3]. For example, Raindsky et al. [2] mined the causality relationships between two events to predict whether a target event will happen after an existing event. They cannot predict unknown events which may not exist in the training data. The other line of works aims to predict the next event by automatically generating a short text describing the next probable event. Hu et al. [4] proposed a CH_LSTM framework for event prediction in a generative way. However, the results are still not satisfactory. They fail to pay attention to critical information at both word level and event level for prediction. In addition, all these existing works do not consider to jointly learn the task of event prediction with other relevant tasks such as event detection, which aims to classify an event to a predefined type.

Event detection and prediction are highly related tasks and can form a two-stage pipeline, in which the stages are intimately correlated. For example, the words indicating the types of the events in event detection should also be scored high in event prediction. Furthermore, the type information of existing events is helpful for next event prediction.

* Corresponding author.
E-mail address: wubin@bupt.edu.cn (B. Wu).

https://doi.org/10.1016/j.neucom.2020.05.023
0925-2312/© 2020 Elsevier B.V. All rights reserved.

In this paper, we investigate a neural attention model that jointly detects the types of events and predicts the next probable event of a sequence of existing events. Our model takes the descriptions (e.g., news titles) of previous events as input and outputs the types of these events as well as generates a short text describing the next probable event. To fully promote the two sub-tasks, we apply neural stacking [5] to the pipeline, feeding the hidden neural layers of the event detection model as additional input features to the event prediction model, and propagating the errors of event prediction to event detection during training, so that information is better shared between the predecessor detection and successor prediction. To improve the performance, we also propose a hierarchical attention mechanism including word-level attention and event-level attention to capture important information for next event prediction. The word-level attention captures the key words which not only indicates the types of the events but also correlates with the next event. On the other hand, the event-level attention mechanism pays attention to important previous events for next event prediction.

Overall, our main contributions can be summarized as follows.

1) To the best of our knowledge, we are the first to consider the relatedness of event prediction and event detection, and propose a novel neural model for joint event detection and prediction, which benefits both the tasks.
2) We present a hierarchical attention mechanism to capture important information at both word level and event level for next event prediction, improving the performance of both event detection and prediction.
3) Empirical experiments reveal that our model achieves substantial improvements on event prediction, advancing several state-of-the-art models. Additionally, the task of event detection also benefits from joint learning.

The remainder of this paper is organized as follows. In Section 2, we define some concepts as well as the problem. In Section 3, we detail our proposed joint model for event detection and prediction. Section 4 describes our experimental results. In Section 5, we review the related literature, followed by conclusion and future research directions in Section 6.

2. Preliminaries

We first define some concepts and the problem of joint event detection and prediction.

Event. An *event* is a particular thing which happened at a specific time and place [1]. In this paper, we consider that the title of a news article describes an event e_m=($w_{m,1},\dots,w_{m,N_m}$), where $w_{m,n} \in \mathscr{V}$ denotes the *n-th* word, and $\mathscr{V}$ denotes the vocabulary.

Event Sequence. An event could typically trigger a sequence of following events. We can denote an event sequence as $s = (e_1, e_2, \dots, e_m)$.

Event Detection. In this paper, the event detection is defined as classifying each event e to a predefined type $t \in T$ (e.g., sports, politics and entertainment).

Event Prediction. Given a sequence of historical events $s = \{e_1, \dots, e_{m-1}\}$ where each event can be represented by a news title e_i=($w_{i,1},\dots,w_{i,N_i}$), event prediction aims to predict the next probable event e_m by generating a short text describing the next probable event. Formally, it can be defined as a language modeling problem:

$$P(e_m|e_{1:m-1}) = \Pi_{n=1}^{N_m} P(w_{m,n}|w_{m,1:n-1}, e_{1:m-1}). \quad (1)$$

So far, tens of thousands of news series containing sequences of events have been recorded as the thing happens, progresses and ends. Reasoning these news series may show us some common patterns about how a typical event sequence developed. For example, in both *earthquake* events and *flood* events, there are sequential events *rescue effort*, *food scarcity*, *chaos* and so on. With the large scale historical data, we can automatically predict the future event given a sequence of observed events by mining the underlying sequential transition patterns.

Joint Event Detection and Prediction. Given a sequence of previously happened events $s = \{e_1, \dots, e_{m-1}\}$, we jointly detect the types of the events $t_1, \dots, t_{m-1}$ and predict the next probable event e_m.

3. Joint model for event detection and prediction

In this section, we present the proposed neural model for **J**oint **E**vent **D**etection and **P**rediction (JEDP) in detail. As illustrated in Fig. 1, our model JEDP consists of two sub models: event detection network and event prediction network, which are based on shared event representation. We stack the two sub models by feeding the hidden neural layers of the event detection network as additional input features to the event prediction network. During training, the errors of event prediction can thus be propagated to event detection, so that information is better shared between the predecessor detection and successor prediction. Both the sub models are based on the shared event representation.

In the following, we will first introduce the shared event representation learning. Then we describe event detection network which classifies the events to predefined types. Finally, we detail the successor sub model for next event prediction with a novel hierarchical attention mechanism, capturing the important information at both word level and event level.

3.1. Shared event representation

We apply a standard Long Short-Term Memory (LSTM) model [6] to learn the shared event representation between different tasks. Let $e_m = (w_{m,1}, w_{m,2}, \dots, w_{m,N_m})$ be an event (note that each event is padded with an *END* token in the end to mark the endings of an event), the LSTM encoder reads the words within the event sequentially and updates its hidden state iteratively. The encoder calculates the hidden vector $\mathbf{h}^w_{m,n}$ at each word position as follows:

$$\mathbf{h}^w_{m,n} = \text{LSTM}(\mathbf{h}^w_{m,n-1}, \mathbf{w}_{m,n}), \quad n = 1, \dots, N_m \quad (2)$$

in which *LSTM* refers to the standard LSTM function[6], $\mathbf{h}^w_{m,n}$ indicates the hidden vector generated at the *n-th* word in *m-th* event. The initial hidden state of LSTM network is set to zero $\mathbf{h}^w_{m,0} = \{0\}$, the same with the initial word state $\mathbf{w}_{m,0} = \{0\}$. $\mathbf{w}_{m,n}$ refers to the input embeddings of word tokens. After consuming the last word of a given event, the hidden state $\mathbf{h}^w_{m,N_m}$ is supposed to capture order-sensitive information within a typical event. We consider it as the final event representation $\mathbf{e}_m$. Note that we can also use other alternative RNN models, such as GRU [7] and Bi-LSTM [6], we do not discuss this since it is not our point in this paper.

3.2. Event detection

Event detection is a multi-class classification task, which can be dealt with a multi-layer perceptron. Formally, given the input vector of an event $\mathbf{e}_m$, a hidden layer is used to induce a set of high-level features $\mathbf{H}_m$:

$$\mathbf{H}_m = \sigma(\mathbf{W}\mathbf{e}_m + \mathbf{b}) \quad (3)$$

Afterwards, $\mathbf{H}_m$ is used as inputs to a softmax output layer:

$$\mathbf{P} = \text{softmax}(\mathbf{W}'\mathbf{H}_m + \mathbf{b}') \quad (4)$$

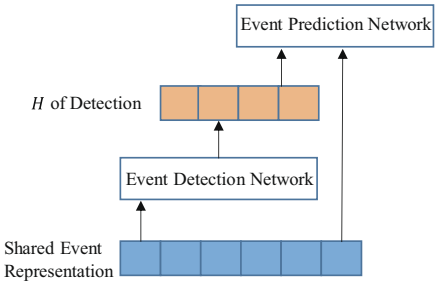


Fig. 1. Illustration of our JEDP model. JEDP consists of two sub models: event detection network and event prediction network. They are based on the shared event representation. The sub models are stacked to enable information sharing between the two tasks.

Here, $\mathbf{W}$, $\mathbf{W}'$, $\mathbf{b}$, $\mathbf{b}'$, are model parameters. $\mathbf{P}$ denotes the probabilities of e_m belonging to each type.

3.3. Next event prediction

Event prediction is a highly related successor subtask of event detection since the salient information are universal between the two tasks. The detected event types are also beneficial for next event prediction. For better integration between event detection and event prediction, we additionally feed the hidden feature vector $\mathbf{H}_m$ of event detection with event embeddings to the event prediction network, as shown in Fig. 2. We first introduce the event sequence encoder which applies another LSTM to project the existing event sequence into a fixed-length embedding. Then we present the hierarchical attention mechanism to capture important information at both word level and event level for improving next event prediction. Finally, we introduce the next event decoder which generates a short text from the sequence encoder, describing the next probable event.

3.3.1. Event sequence encoder

The event sequence encoder takes the representations of existing events $(\mathbf{e}_1, \mathbf{e}_2, \dots \mathbf{e}_{m-1})$ as well as the hidden vectors $(\mathbf{H}_1, \mathbf{H}_2, \dots, \mathbf{H}_{m-1})$ of event detection as input and calculates a sequence of recurrent states:

$$\mathbf{h}^e_m = \mathrm{LSTM}(\mathbf{h}^e_{m-1}, \mathbf{e}_m \oplus \mathbf{H}_m), \tag{5}$$

where $\mathbf{e}_m \oplus \mathbf{H}_m$ is a concatenation of the event embedding $\mathbf{e}_m$ and the hidden state of event detection for the event $\mathbf{H}_m$. $\mathbf{H}_m$ offers information about the type of the event, which is useful for predicting the next event. It connects the detection and prediction steps, so that information sharing is enhanced between them and back propagation is enabled for upgrading all the model parameters. We set $\mathbf{h}^e_0 = \{0\}$, $\mathbf{e}_0 = \{0\}$ for initialization. The sequence encoder computes the current hidden state $\mathbf{h}^e_m$ after consuming the current m-th event and the hidden state of the previous time step $\mathbf{h}^e_{m-1}$, and thus updates its internal state iteratively. After consuming all the events in the sequence, the last hidden state is taken as the final representation of the event sequence, which is believed to contain the information of all the observed events.

3.3.2. Hierarchical attention mechanism

To consider the different importance of information at both word level and event level, we propose a hierarchical attention mechanism including word-level attention which puts different weights on words within an event and event-level attention which quantifies the importance of previous events within a sequence. By incorporating the importance of both individual words and events, we are supposed to get better results of event prediction.

Word-level attention. The word-level attention is designed here to encourage the model to stress valuable words in the input sequence, instead of limiting to the last few words which is a severe constraint suffered by the LSTM models [6]. It forces the model to attend over specific parts over input word sequences. By linking the output of current decoding step with word tokens in previous input events, this attention mechanism highlights the words which play a key role in next event generation.

As demonstrated in Fig. 2, during event representation learning, the LSTM network reads word sequences and generates a hidden state for each word. The strength indicator $\mathbf{a}_{t,(m,n)}$ is calculated between the hidden state $\mathbf{h}^{w'}_t$ at the current decoding step t and the hidden word state $\mathbf{h}^w_{m,n}$ of the word $w_{m,n}$. Formally,

$$a^w_{t,(m,n)} = \mathbf{U}^{\mathrm{T}} f(\mathbf{W}_1 \cdot \mathbf{h}^{w'}_t + \mathbf{W}_2 \cdot \mathbf{h}^w_{m,n}) \tag{6}$$

We penalize the input words that have already obtained high scores for generating a certain word in the decoding step according to

$$a'^w_{t,(m,n)} = \begin{cases} exp(\mathbf{a}^w_{t,(m,n)}) & if\ t = 1 \\ \frac{exp(a^w_{t,(m,n)})}{\sum_{j=1}^{t-1} exp(a^w_{j,(m,n)})} & otherwise \end{cases} \tag{7}$$

Then, the normalized attention weight $\alpha^w_{t,(m,n)}$ is calculated across all the words in the input events.

$$\alpha^w_{t,(m,n)} = \frac{a'^w_{t,(m,n)}}{\sum_{i=1}^{M}\sum_{j=1}^{N_i} a'^w_{t,(i,j)}}, m \in [1, M], n \in [1, N_m] \tag{8}$$

In the end, the word-level attention vector at each decoding step can be calculated by adding the multiplications of attention weights with corresponding word hidden vectors. The attention vector is believed to contain the information about key words for next event prediction. Formally,

$$\mathbf{att}^w_t = \sum_{m=1}^{M}\sum_{n=1}^{N_i} \alpha^w_{t,(m,n)} \cdot \mathbf{h}^w_{m,n} \tag{9}$$

Event-level attention. Similarly, event-level attention aims to suggest which events are more responsible for next event prediction. Specifically, at current decoding step t, the event-level strength indicator a^e_m is calculated as follows.

$$a^e_{t,m} = \mathbf{U}'^{\mathrm{T}} f(\mathbf{W}'_1 \cdot \mathbf{h}^{w'}_t \mathbf{W}'_2 \cdot \mathbf{h}^e_m) \tag{10}$$

We penalize events that have already obtained high scores for generating a certain word in previous decoding steps. Formally,

$$a'^e_{t,m} = \begin{cases} exp(a^e_{t,m}) & if\ t = 1 \\ \frac{exp(a^e_{t,m})}{\sum_{j=1}^{t-1} exp(a^e_{j,m})} & otherwise \end{cases} \tag{11}$$

The normalized attention weight $\alpha^e_{t,m}$ is calculated across all the events in input sequence.

$$\alpha^e_{t,m} = \frac{a'^e_{t,m}}{\sum_{i=1}^{M} a'^e_{t,i}}, m \in [1, M] \tag{12}$$

We compute the event-level attention vector $\mathbf{att}^e_t$ indicating which events play a critical role for next event prediction, at each decoding step t:

$$\mathbf{att}^e_t = \sum_{m=1}^{M} \alpha^e_{t,m} \cdot \mathbf{h}^e_m \tag{13}$$

3.3.3. Next event decoder

After encoding the sequence of events $(e_{1:m-1})$, an LSTM decoder is designed to interpret the compressed information into word

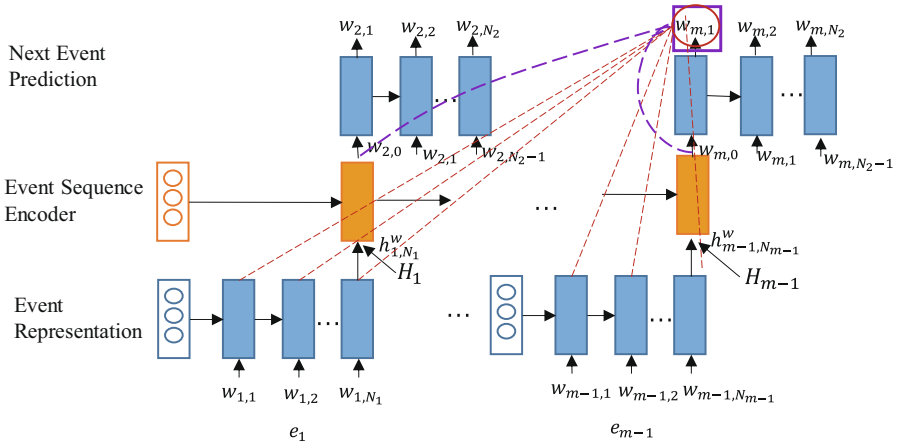


Fig. 2. Illustration of event prediction network. First, the shared event representations and the hidden feature vector $\mathbf{H}_m$ of event detection are fed together into the event sequence encoder to obtain sequence representation. Then, a hierarchical attention mechanism including word level attention (shown in red line) and event level attention (in purple) is incorporated with the sequence representation to capture critical information for next event prediction. Finally, a next event decoder is used to generate a short text describing the next possible event. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

tokens describing the next probable event e_m. At each decoding step t, the hidden state of the decoder is updated with:

$$\mathbf{h}_t^{w'} = \text{LSTM}(\mathbf{h}_{t-1}^{w'}, \mathbf{w}_{t-1}) \tag{14}$$

The initial hidden state of the decoder, $\mathbf{h}_0^{w'} = h_{m-1}^e$. The initial word state $\mathbf{w}_0 = \{0\}$. In the training process, the decoder is fed with the ground-truth word tokens in the next event, $(w_1, w_2, \ldots w_t, \ldots)$ [8].

Incorporating Hierarchical Attention. To pay attention to valuable words and events, we further incorporate the hierarchical attentive information by concatenating the output of decoder $\mathbf{h}_t^{w'}$ with the dual-level attention vectors, namely word-level attention vector $\mathbf{att}_t^w$ and event level attention vector $\mathbf{att}_t^e$ at each decoding step. Afterwards, a Softmax layer is used to generate the final distributions of word tokens at each decoding step t based on:

$$\mathbf{P}(w_t) = \text{Softmax}(\hat{\mathbf{W}}[\mathbf{h}_t^{w'} \| \mathbf{att}_t^w \| \mathbf{att}_t^e] + \hat{\mathbf{b}}) \tag{15}$$

Instead of sampling by greedy strategy, we adopt random sampling technique according to the probability distribution $\mathbf{P}(w_t)$. At test time, our model terminates until generating the "*END*" symbol.

3.4. Model training

Our training objective is to minimize the cross-entropy loss and negative log likelihood corresponding to the tasks of event detection and event prediction, respectively. Formally, given existing events $e_{1:m-1}$, the objective function is defined as follows:

$$L = -\sum_{s=1}^{S_{train}} \sum_{m=2}^{M} \left(\sum_{i=1}^{m-1} y_i log(y_i) + logP(e_m^s | e_{1:m-1}^s) \right) \tag{16}$$

where S_{train} is the total number of news series each containing a sequence of events, $y_i \in T$ denotes the true type of the event e_i. We get the optimal model parameters θ^* by minimizing the L. ADAM [9] was adopted for optimization.

4. Experiments

In this section, we validate the effectiveness of our proposed model through experiments on a real-world dataset.

4.1. Dataset

We evaluate our proposed model JEDP on a large-scale real-world **Chinese Sina News Series** dataset, which is the same dataset used in [4]. The dataset contains 15,254 news series, and each series includes about 50 news articles in average (the title of a news article is regrated as the description of an event in this paper). They cover 15 event types including politics, sports and so on. Following [4], we sort the news articles in chronological order and use a window of size 5 to segment the news articles to get non-overlapping news sequences. Finally, we obtain order-sensitive news sequence set, containing 155,358 news sequences in total. We adopt JIEBA tool[1] to perform Chinese word segmentation. We remove the stopwords and further prune the vocabulary V by dropping the words that occur less than 100 documents. Finally, we get a vocabulary of size 8,107, including an "*END*" symbol.

We randomly split our dataset into three parts, 80% for training set, 10% for testing set and 10% for validation set. In detail, the training set contains 124,288 news series, and 607,090 events. There are 15,535 news series and 75,802 events in the validation set. While we have 15,535 news series, with corresponding 75,957 events in the test set.

4.2. Implementation details

Baselines. Six state-of-the-art baseline models are implemented here to demonstrate the effectiveness of our proposed model. For all the n-gram language models, we choose $n = 5$. In addition, we also implement two variants of our proposed model JDEP to study the effect of the hierarchical attention mechanism and reinforcement learning separately.

- Backoff N-gram [10]: An N-gram language model using backoff smoothing.
- modified Kneser-Key [10]: An N-gram language model using Kneser-key smoothing.
- Witten-Bell Discounting N-Gram [10]: An N-gram language model using Witten-Bell Smothing.
- LSTM [4]: This model is implemented by the basic LSTM sequence, which treats all the words in previous events as a whole sequence, and generates next probable event after con-

[1] https://pypi.org/project/jieba/.

suming all the words in previous events.
- HLSTM [4]: A hierarchical LSTM model which is implemented with dual-level encoder and a next event predictor.
- CH_LSTM [4]: A model incorporating topic information into HLSTM model.
- JEDP w/o Att: A variant of our proposed JEDP model without the hierarchical attention mechanism.
- JEDP w/o Joint: A variant of our proposed JEDP model without joint learning of event detection and prediction.

We set the dimension of LSTM hidden state and word embeddings as 300D for all of the models discussed herein. The word embeddings are uniformly initialized with a range of $[-0.8, 0.8]$ and updated during training. The learning rate is initialized as 0.0005 and the batch size is set as 32. To avoid over-fitting, we set the dropout rate to 0.2. We evaluate the model on the validation set and select the model with the best BLEU score as our model.

Evaluation Metrics. We choose two metrics, the perplexity [11] and the BLEU to evaluate the effectiveness of our model for next event prediction.

Perplexity is a standard metric in information theory [12]. It measures how well a model fits the data and thus can perform better prediction. Lower perplexity indicates a better model. Formally, the per-word perplexity of a model is defined as follows.

$$Perplexity = exp\left(-\frac{1}{N_w}\sum_{s=1}^{S_{train}} logP(e_m^s|e_{1:m-1}^s\right) \quad (17)$$

To further analyse the readability and coherency correlated with human judgement, we adopt BLEU score to evaluate the generated results. BLEU is proposed to automated understudy to skilled human judges, which claims a high correlation with human judgements of quality [13]. Specifically, we adopt standard BLEU. Higher scores on these metrics indicate better performance of the models.

For event detection which classifies each event to predefined types, we use the metric of accuracy to evaluate the performance.

4.3. Experiment results

We report the performance of different models for event prediction on the Chinese Sina News Series dataset in Table 1. We can see that all neural network based models significantly outperform traditional n-gram language models by a large margin. Among all the methods, our proposed JEDP model yields the best performance, reducing the perplexity by around 40 % and improving the BLEU score by around 21% compared to the state-of-the-art method CH_LSTM.

4.4. Comparison of variants of JEDP

We also compare our model with several variants to validate the effectiveness of our proposed hierarchical attention mechanism and joint modeling. As we can see from Table 2, without attention, the performance of our model drops significantly on both event detection and event prediction. It demonstrates that the hierarchical attention can capture important information at both word level and event level for event prediction, which further benefits the joint task, i.e. event detection. We can also see that removing joint modeling also depresses the performance on both tasks, especially on event prediction.

4.5. Illustration of hierarchical attention

To further illustrate the effect of the proposed hierarchical attention mechanism, as shown in Fig. 3, we present two heat maps to visualize what information the proposed model JEDP put emphasis on during the decoding process. The top of the figure shows the attention of JEDP on the previous events and words when decoding the first word "France". The bottom shows the attention of JEDP on the events and words when decoding the last word "Chaos". As we can see from the left bar, when generating the first word "France", the model pays attention to the events e_1, e_3. Specifically, the model stresses on the word tokens including "France", "Paris", and "National". While decoding the last word "Chaos", the model pays more attention to the events e_1, e_2. Particularly, it emphasizes the word tokens, "Explosion", "Hostage", "Emergency" and "Terror". The heat maps show that by incorporating the hierarchical attention mechanism, our JEDP model can capture valuable information from both word-level and event-level to improve next event prediction.

Table 1
Comparison of different models on event prediction. We report the results in terms of Perplexity and BLEU.

Model	Perplexity	BLEU
Backoff N-gram [10]	884	9.1
Modified Kneser-Ney [10]	870	9.3
Witten-Bell Discounting N-Gram [10]	835	9.1
LSTM [14]	588	21.3
HLSTM [4]	526	22.3
CH_LSTM [4]	483	24.5
JEDP (ours)	293	29.7

Table 2
Comparison of our model with variants.

Model	Perplexity	BLEU	Accuracy
JEDP w/o Att	427	25.1	0.86
JEDP w/o Joint	316	28.3	0.93
JEDP	293	29.7	0.95

4.6. Case study

In this subsection, we provide a qualitative analysis for the generated output events of the models. We take a news series about "11.13 Paris Terror Attack" for example. As shown in Table 3, we list the previous observed events, the next event (ground-truth) and the predicted events generated by the models. We compare our JEDP with state-of-the-art model CH_LSTM and two variants of JEDP (i.e., without Att or without Joint).

From Table 3, we can see that, for news series "11.13 Paris Terror Attack" case, the next event generated by the state-of-the-art model CH_LSTM contains limited information since it only capture two key concepts the "France" and "Terror Attack". Our model and its variants provide more plentiful information. Furthermore, the next event generated by our model JEDP, predicting the "chaos", is most consistent with the ground-truth.

It demonstrates the effectiveness of our model with the hierarchical attention mechanism and joint modeling of event detection.

4.7. Next event ranking

Following [4], we also evaluate our model on next event ranking task. Given a news series, this task is to find the most probable next event within in a candidate set. Ideally, our model is expected to assign the ground-truth event with the highest probability within the candidate set.

Detailed Process. To obtain the dataset for this task, we merge the validation set and the test set, containing 31,070 news series in total. Following [4], we randomly divided the dataset (31,070 news

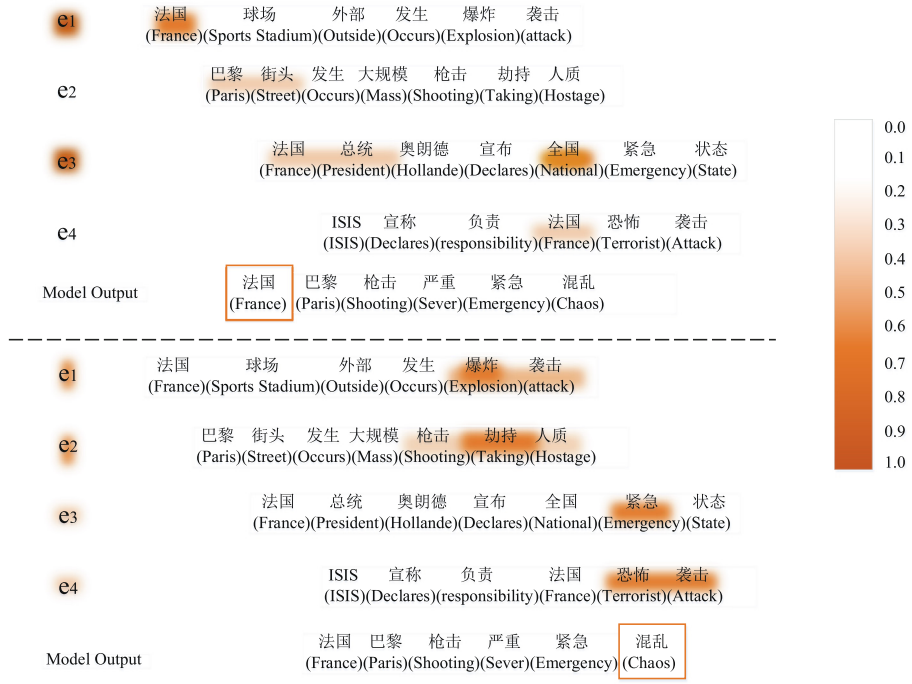


Fig. 3. Heat maps illustrating the attention of our proposed JEDP on previous events and words for prediction.

Table 3
An example of generated outputs of different models for the news series "11.13 Paris Terror Attack" given the existing events $e_{1:4}$.

Previous Events in a News Series $(e_1 : e_{m-1})$	e_1: 法国球场外部发生爆炸袭击 (France)(Sports stadium)(Outside)(Occurs)(Explosion)(Attack)
	e_2: 巴黎街头发生大规模枪击劫持人质 (Paris)(Street)(Occurs)(Mass)(Shooting)(Hostage-taking)
	e_3: 法国 总统 奥朗德 宣布 全国 紧急 状态 (France)(President)(Hollande)(Declares)(National)(Emergency)(State)
	e_4: ISIS 宣称负责法国恐怖袭击 (ISIS)(Declares)(Responsible)(France)(Terror)(Attack)
Next Event e_m (Ground Truth)	恐怖袭击法国造成伤亡混乱 (Terror)(Attack)(France)(Cause)(Injuries)(Chaos)
CH_LSTM Output	法国称袭击恐怖袭击 (France)(Claim)(Attack)(Terror Attack)
JEDP w/o Att Output	法国袭击图 (France)(Attack)(Figure)
JEDP w/o Joint Output	法国枪击已致人失踪 (France)(Shooting)(Cause)(People)(Missing)
JEDP Output	法国巴黎枪击严重紧急混乱 (France)(Paris)(Shooting)(Sever)(Emergency)(Chaos)

series) into 621 non-overlapping subsets with each containing 50 news series except the last one which contains remaining 20 series (i.e., 31,070 = 621 $*$ 50 + 20). For each news series (consisting of a few events) in a subset, we aim to choose the best last event given its previous events. The candidate set is composed of the last events of all the series in the corresponding subset. We use the measurement *his@n* which indicates the BKI8probability of the correct events within the top *n* ranked events.

Result Analysis. Table 4 illustrates the performance of our model compared with state-of-the-art neural models, in terms of

hits@1, hits@5, hits@10. The results are consistent with Table 1. Our models and variants all outperforms existing neural models, which shows the effectiveness of joint modeling and hierarchical attention. It is worth noting that our model JEDP has significant improvement (around 2%) compared to the baseline models in terms of the most important metric hits@1. Compared to the variants without attention or joint modeling, our model achieves the best performance. In summary, the results demonstrate the effectiveness of our joint model JEDP with a hierarchical attention mechanism.

Table 4
The performance of different models on next event ranking task in terms of *hits*@1, *hits*@5, *hits*@10.

Model	hits@1	hits@5	hits@10
LSTM	23.04	51.23	67.36
HLSTM	26.99	56.97	71.23
CH_LSTM	28.03	57.12	72.34
JEDP w/o ATT	28.10	57.53	72.81
JEDP w/o Joint	29.54	58.03	73.51
JEDP	29.89	58.11	73.90

5. Related work

In this section, we review the related literature. Our work is mainly related to event detection and event prediction.

5.1. Event detection

Event detection in the Topic Detection and Tracking (TDT) program [1] aims to discover new or previously unidentified events in an unsupervised manner. Many works focused on detecting events of interest from social media [15,16] since social media is more instant. A lot of researches on ACE event detection task extracted events with entities from sentences [17–19]. Different from these event detection tasks, we study the problem of categorizing each event (represented by a news title) to predefined types, which can be formalized as a text classification problem.

Traditional text classification methods such as SVM [20] are based on feature engineering. The most commonly used features are BoW and TF-IDF [21]. With the proliferation of neural networks, neural models have been successfully applied in text classification [22]. Two representative deep neural models such as RNNs [23] and CNNs [24,25] have shown their power for text representation in many NLP tasks, including text classification and so on.

5.2. Event prediction

Although great effort has been dedicated to event detection based on social media [26,27] and search engines [28], a relatively few works have been proposed to predict future events.

The work on event prediction can be divided into two categories. On one hand, some work learn the causal relations of two events for prediction [2]. For instance, Radinsky et al. [2] extracted generalized causality relations of two events (i.e., "x causes y") from past news and applied them to predict the next possible event given a current event. Granroth-Wilding et al. [3] extracted typical sequences of events from texts [29] and used a compositional neural network to learn the coherence score of two events. They aim to predict the next event by learning the strength of association between two event.

On the other hand, some work focus on mining event sequence pattern for prediction. For example, Radinsky et al. [30] extracted event chains from news documents for predicting the happening of target events. In this work, we focus on non-targeted event prediction. Along this line, Manshadi et al. [31] learned a *n*-gram language model of event sequences from Internet Web log stories. Pichotta and Mooney [14] developed a LSTM based model for learning scripts which represents knowledge of prototypical event sequences. They represented an event as a predicate with several arguments and are limited to predict the events from candidates. Hu et al. [4] proposed a context-aware hierarchical LSTM prediction model which directly learns event representations and generate next event with a sequence to sequence network.

In this work, we propose a novel neural model for joint event detection and prediction. In our model, we apply a new hierarchical attention mechanism to capture important information at both word level and event level.

5.3. Attention mechanism

The concept of "attention" has gained popularity recently in neural networks, allowing models to learn alignments between different modalities. It has been successfully applied in a wide range of tasks, including image captioning [32], image classification [33] and machine translation [34] etc. Attention mechanism quantifies the related degree of different parts in the input sequence with target sequence in order to compute a representation of the input sequence. A lot of works have adopted attention mechanism to improve the model's performance. For example, Cheng et al. [35] use LSTM and attention mechanism to facilitate the task of machine reading. Sankaran et al. [36] propose an intra-temporal attention on input sequences to prevent the model from attending over same parts of the input during different decoding step. Nallapati et al. [37] prove that this mechanism can alleviate repetition problem in long document summarization. Some works found that purely building models with attention mechanism, dispensing with recurrence and convolutions entirely, can also achieve competitive performance. Vaswani et al. [38] propose the transformer model, which is purely built on attention mechanism and shows powerful performance on machine translation. Shen et al. [39] propose an RNN/CNN-free sentence-encoding model, which introduces the reinforcement learning to hybrid hard-attention with soft attention. Some works have noticed the effect of structured attention. Lin et al. [40] propose an representation learning model for extracting sentence embedding by structured attention. Zhang et al. [41] present a user-guided hierarchical attention network to hierarchically attend both visual and textual modalities.

Differently, we present a novel hierarchical attention mechanism to attend both word-level and event-level information for event prediction task.

6. Conclusion and future work

In this paper, we propose a novel neural attention model JEDP for joint event detection and prediction, which boosts both tasks through information sharing between the two tasks. Our model takes the sequence of previous events as input, and classifies these events to predefined types as well as predicts the next probable event by generating a short text describing it. Additionally, we propose a hierarchical attention mechanism which enables the model to capture important information at both event-level and word-level for predicting the next event. Empirical experiments on a real-world dataset demonstrate the superior performance of our model over state-of-the-art methods on both tasks, especially on event prediction.

In future work, we will explore the effectiveness of our model on other tasks such as document summarization.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

CRediT authorship contribution statement

Linmei Hu: Conceptualization, Methodology, Writing - original draft, Writing - review & editing, Funding acquisition. **Shuqi Yu:** Software, Data curation, Validation. **Bin Wu:** Supervision, Resources, Project administration. **Chao Shao:** Conceptualization, Methodology. **Xiaoli Li:** Writing - review & editing.

Acknowledgements

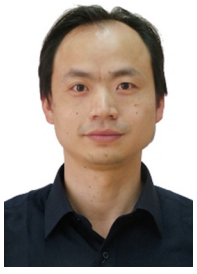
The work is supported by the National Natural Science Foundation of China (Nos. 61806020, 61972047, U1936220), and the Fundamental Research Funds for the Central Universities.

References

[1] J. Allan, R. Papka, V. Lavrenko, On-line new event detection and tracking, in: SIGIR, 1998, pp. 37–45.
[2] K. Radinsky, S. Davidovich, S. Markovitch, Learning causality for news events prediction, WWW (2012) 909–918.
[3] M. Granroth-Wilding, S. Clark, What happens next? Event prediction using a compositional neural network model, in: AAAI, 2016, pp. 2727–2733.
[4] L. Hu, J. Li, L. Nie, X. Li, C. Shao, What happens next? Future subevent prediction using contextual hierarchical LSTM, in: AAAI, 2017, pp. 3450–3456.
[5] H. Chen, Y. Zhang, Q. Liu, Neural network for heterogeneous annotations, in: EMNLP, 2016, pp. 731–741.
[6] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Computation 9 (8) (1997) 1735–1780.
[7] J. Chung, Ç. Gülçehre, K. Cho, Y. Bengio, Empirical evaluation of gated recurrent neural networks on sequence modeling, NIPS..
[8] R.J. Williams, D. Zipser, A learning algorithm for continually running fully recurrent neural networks, Neural Computation 1 (2) (1989) 270–280.
[9] D.P. Kingma, J. Ba, Adam: a method for stochastic optimization, ICLR (Poster)..
[10] A. Stolcke, SRILM - an extensible language modeling toolkit, in: INTERSPEECH, 2002..
[11] Y. Bengio, R. Ducharme, P. Vincent, C. Janvin, A neural probabilistic language model, Journal of Machine Learning 3 (Feb) (2003) 1137–1155..
[12] C.E. Shannon, A mathematical theory of communication, Mobile Computing and Communications Review 5 (1) (2001) 3–55.
[13] K. Papineni, S. Roukos, T. Ward, W. Zhu, Bleu: a method for automatic evaluation of machine translation, in: ACL, 2002, pp. 311–318..
[14] K. Pichotta, R.J. Mooney, Learning statistical scripts with LSTM recurrent neural networks, in: AAAI, 2016, pp. 2800–2806.
[15] X. Zhang, X. Chen, Y. Chen, S. Wang, Z. Li, J. Xia, Event detection and popularity prediction in microblogging, Neurocomputing 149 (2015) 1469–1480.
[16] W. Cui, P. Wang, Y. Du, X. Chen, D. Guo, J. Li, Y. Zhou, An algorithm for event detection based on social media data, Neurocomputing 254 (2017) 53–58.
[17] Q. Li, H. Ji, Incremental joint extraction of entity mentions and relations, in: ACL, 2014, pp. 402–412.
[18] D. Zeng, K. Liu, S. Lai, G. Zhou, J. Zhao, et al., Relation classification via convolutional deep neural network, in: COLING, 2014, pp. 2335–2344..
[19] X. Feng, B. Qin, T. Liu, A language-independent neural network for event detection, Science China Information Sciences 61 (9) (2018) 092106.
[20] H. Drucker, D. Wu, V.N. Vapnik, Support vector machines for spam categorization, IEEE Transactions on Neural networks 10 (5) (1999) 1048–1054.
[21] C.C. Aggarwal, C. Zhai, A survey of text classification algorithms, in: Mining Text Data, Springer, 2012, pp. 163–222..
[22] G. Liu, J. Guo, Bidirectional LSTM with attention mechanism and convolutional layer for text classification, Neurocomputing 337 (2019) 325–338.
[23] P. Liu, X. Qiu, X. Huang, Recurrent neural network for text classification with multi-task learning, in: IJCAI, 2016, pp. 2873–2879..
[24] Y. Kim, Convolutional neural networks for sentence classification, in: EMNLP, 2014, pp. 1746–1751.
[25] P. Wang, B. Xu, J. Xu, G. Tian, C.-L. Liu, H. Hao, Semantic expansion using word embedding clustering and convolutional neural network for improving short text classification, Neurocomputing 174 (2016) 806–814.
[26] T. Sakaki, M. Okazaki, Y. Matsuo, Earthquake shakes twitter users: real-time event detection by social sensors, in: WWW, 2010, pp. 851–860.
[27] L. Mu, P. Jin, L. Zheng, E. Chen, L. Yue, Lifecycle-based event detection from microblogs, in: WWW, 2018, pp. 283–290.
[28] J. Ginsberg, M.H. Mohebbi, R.S. Patel, L. Brammer, M.S. Smolinski, L. Brilliant, Detecting influenza epidemics using search engine query data, Nature 457 (7232) (2009) 1012–1014.
[29] N. Chambers, D. Jurafsky, Unsupervised learning of narrative event chains, in: ACL, 2008, pp. 789–797.
[30] K. Radinsky, E. Horvitz, Mining the web to predict future events, in: WSDM, 2013, pp. 255–264.
[31] M. Manshadi, R. Swanson, A.S. Gordon, Learning a probabilistic model of event sequences from internet weblog stories, in: FLAIRS, 2008, pp. 159–164..
[32] K. Xu, J. Ba, R. Kiros, K. Cho, A.C. Courville, R. Salakhutdinov, R.S. Zemel, Y. Bengio, Show, attend and tell: Neural image caption generation with visual attention, in: ICML, 2015, pp. 2048–2057..
[33] T. Xiao, Y. Xu, K. Yang, J. Zhang, Y. Peng, Z. Zhang, The application of two-level attention models in deep convolutional neural network for fine-grained image classification, in: CVPR, 2015, pp. 842–850..
[34] K. Cho, B. van Merrienboer, Ç. Gülçehre, D. Bahdanau, F. Bougares, H. Schwenk, Y. Bengio, Learning phrase representations using RNN encoder-decoder for statistical machine translation, in: EMNLP, 2014, pp. 1724–1734.
[35] J. Cheng, M. Lapata, Neural summarization by extracting sentences and words, in: ACL, 2016.
[36] B. Sankaran, H. Mi, Y. Al-Onaizan, A. Ittycheriah, Temporal attention model for neural machine translation, arXiv preprint arXiv:1608.02927..
[37] R. Nallapati, B. Zhou, C.N. dos Santos, Ç. Gülçehre, B. Xiang, Abstractive text summarization using sequence-to-sequence rnns and beyond, in: CoNLL, 2016, pp. 280–290.
[38] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, L. Kaiser, I. Polosukhin, Attention is all you need, in: NIPS, 2017, pp. 5998–6008..
[39] T. Shen, T. Zhou, G. Long, J. Jiang, S. Wang, C. Zhang, Reinforced self-attention network: a hybrid of hard and soft attention for sequence modeling, in: IJCAI, 2018, pp. 4345–4352..
[40] Z. Lin, M. Feng, C.N. dos Santos, M. Yu, B. Xiang, B. Zhou, Y. Bengio, A structured self-attentive sentence embedding, in: ICLR (poster), 2017.
[41] W. Zhang, W. Wang, J. Wang, H. Zha, User-guided hierarchical attention network for multi-modal social image popularity prediction, in: WWW, 2018, pp. 1277–1286.

Linmei Hu received her Ph.D. degree from Tsinghua University, China in 2018. She is currently an assistant professor at School of Computer Science, Beijing University of Posts and Telecommunication, China. Her research interests are text mining and knowledge graph.

Shuqi Yu is expected to receive master degree of Computer Science from Beijing University of Posts and Telecommunications in June 2020. Her research interests are in natural language processing, text mining, event analysis.



Bin Wu received his Ph.D. degree from the Institute of Computing Technology, Chinese Academy of Science, China in 2002. He is a senior member of CCF. He is currently a professor at the School of Computer Science, Beijing University of Posts and Telecommunication, China. His research interests are in data science, complex network, and big data technology. He has published more than 100 papers in referred journals and conferences.

Chao Shao received his master degree from the Department of Computer Science and Technology, Tsinghua University. He is currently an engineer of didi company. His research interests include recommendation, intelligent marketing algorithm, and machine learning.

Dr. Xiao-li Li is currently a department head (Data Analytics department, consisting of 70+ data scientists, which is Singapore largest data analytics group) and a senior scientist at the Institute for Infocomm Research, A*STAR, Singapore. He also holds adjunct associate professor positions at National University Singapore and Nanyang Technological University.