Extending Event Detection to New Types with Learning from Keywords

Viet Dac Lai and **Thien Huu Nguyen**
Department of Computer and Information Science
University of Oregon, OR, USA
{vietl,thien}@cs.uoregon.edu

Abstract

Traditional event detection classifies a word or a phrase in a given sentence for a set of predefined event types. The limitation of such predefined set is that it prevents the adaptation of the event detection models to new event types. We study a novel formulation of event detection that describes types via several keywords to match the contexts in documents. This facilitates the operation of the models to new types. We introduce a novel feature-based attention mechanism for convolutional neural networks for event detection in the new formulation. Our extensive experiments demonstrate the benefits of the new formulation for new type extension for event detection as well as the proposed attention mechanism for this problem.

1 Introduction

Event detection (ED) is a task of information extraction that aims to recognize event instances (event mentions) in text and classify them into specific types of interest. Event mentions are usually associated with an event trigger/anchor in the sentence of the event mentions, functioning as the main word to evoke the event. For instance, in the sentence *"She is going to leave to become chairman of Time Inc."*, an ED system should be able to recognize that the word "*leave*" is triggering an event of type "*End-Position*".

There have been two major approaches for ED in the literature. The first approach focuses on the development of linguistic features to feed into the statistical models (i.e., MaxEnt) (Ahn, 2006; Ji and Grishman, 2008; Liao and Grishman, 2010; McClosky et al., 2011). The second approach, on the other hand, relies on deep learning (i.e., convolutional neural networks (CNN)) to automatically induce features from data (Chen et al., 2015; Nguyen et al., 2016a; Liu et al., 2017; Lu and Nguyen, 2018), thus significantly improving the performance for ED.

One limitation of the current approaches for ED is the assumption of a predefined set of event types for which data is manually annotated to train the models. For example, the popular benchmark dataset ACE 2005 for ED annotates 8 types and 33 subtypes of events. Once the models have been trained in this way, they are unable to extract instances of new, yet related types (i.e., having zero performance on the new types). To extend the operation of these models into the new types, the common approach is to spend some effort annotating data for the new types to retrain the models. Unfortunately, this is an expensive process as we might need to obtain a large amount of labeled data to adequately represent various new event types in practice. Such expensive annotation has hindered the application of ED systems on new types and calls for a better way to formulate the ED problem to facilitate the extension of the models to new event types.

In this paper, we investigate a novel formulation of ED where the event types are defined via several keywords instead of a large number of examples for event types in the traditional approaches (called the learning-from-keyword formulation (LFK)). These keywords involve the words that can possibly trigger the event types in the contexts. For instance, the event type *End-Position* can be specified by the keywords ("*left*, "*fired*", "*resigned*"). Given the keywords to represent event types, the ED problem becomes a binary classification problem whose goal is to predict whether a word in a sentence expresses the event type specified by the keywords or not. This formulation enables the ED models to work with new event types as long as the keywords to describe the new types are provided, thus allowing the ED models to be applicable on a wide range of

Proceedings of the 2019 EMNLP Workshop W-NUT: The 5th Workshop on Noisy User-generated Text, pages 243–248
Hong Kong, Nov 4, 2019.

new event types and mitigating the needs for large amounts of annotated data for the new types.

The goal of this paper is to evaluate the effectiveness of LFK in the new type extension setting for ED where the models are trained on labeled data from some types but applied to extract instances of unseen types. We would like to promote this problem as a new task for ED for future research. To set the baselines for this problem, we employ the ACE 2005 dataset and recast it into LFK. We examine the performance of the baseline models for ED in the traditional formulation when they are adapted to LFK. The experiments show that with the new formulation, such ED models can actually recognize new event types although their performance should be still further improved in future research. Finally, we demonstrate one possibility to improve the performance of the baseline ED models in LFK by presenting a novel attention mechanism for CNNs based on the feature space to fuse the representations of the keywords and contexts. We achieve the state-of-the-art performance for the new type extension with the proposed attention mechanism.

2 Related work

In the last decade, many machine learning systems have been introduced to solve ED. Before the era of the deep neural networks, these systems are mainly based on supervised learning using extensive feature engineering with the machine learning frameworks (Ahn, 2006; Ji and Grishman, 2008; Hong et al., 2011; Riedel et al., 2009; Riedel and McCallum, 2011a,b; Miwa et al., 2014; Li et al., 2014, 2015). Recently, many advanced deep learning methods were introduced to enhance event detectors such as distributed word embedding (Chen et al., 2015; Nguyen et al., 2016b; Liu et al., 2017; Nguyen and Nguyen, 2019), convolutional neural networks (Chen et al., 2015, 2017; Nguyen and Grishman, 2015; Nguyen et al., 2016b; Nguyen and Grishman, 2018), recurrent neural networks (Nguyen et al., 2016b; Sha et al., 2018), and the attention mechanism (Liu et al., 2017; Nguyen and Nguyen, 2018b; Liu et al., 2018). However, the models proposed in these work cannot extend their operation to new event types.

Regarding the new formulations for ED, previous studies(Bronstein et al., 2015; Peng et al., 2016) also examine keywords to specify event types. However, these studies do not investigate the new type extension setting as we do in this. Recently, zero-shot learning is employed for new types in event extraction(Huang et al., 2018); however, the event types are specified via the possible roles of the arguments participating into the events in this work. It also uses complicated natural language processing toolkits, making it difficult to apply and replicate the settings. Our work emphasizes the simplicity in the setting for new type extension to facilitate future research. Finally, extending ED to the new type is investigated using real examples as new event types (Nguyen et al., 2016c). However, it requires a large number of examples to perform well. Our work instead requires only a few keywords to help the models achieve reasonable performance on new types.

3 Learning-from-Keywords for ED

3.1 Task Definition

In the learning-from-keyword formulation for ED, the inputs include a context (i.e., an n-word sentence $X = \{x_1, x_2, \dots, x_n\}$ with an anchor word located at position a (the word x_a)) and a set of keywords K. The words in K are the possible trigger words of some event type of interest. The goal is to predict whether the word x_a in S expresses the event type specified by K or not (i.e., a binary classification problem to decide whether the context matches the event keywords or not). An example in LFK thus has the form (X, x_a, K, Y) where Y is either 1 or 0 to indicate the match of X and K.

3.2 Data Generation

To facilitate the evaluation of the ED models in LFK for the new type extension setting, we need to obtain training and test/development datasets so the keyword sets of the examples in the test/development datasets define event types that are different from those specified by the keyword sets in the training datasets. To our best knowledge, there is no existing data following LFK setting, therefore, in this section, we present a process to automatically generate an ED dataset for LFK setting from an existing ED dataset.

We obtain these datasets by leveraging ACE 2005, the popular benchmark datasets for ED. ACE 2005 dataset is annotated for 8 event types $\mathcal{T} = \{t_1, t_2, \dots, t_8\}$, and 33 event subtypes $\mathcal{S} = \{s_1, s_2, \dots, s_{33}\}$. There is also a special type/subtype of "*Other*" indicating the non-event instances (*Other* $\notin \mathcal{S}$) . As each event subtype in ACE 2005 is associated with one event type, let $\mathcal{C}_i$ be the set of subtypes corresponding to the type $t_i \in \mathcal{T}$. Also, let $\mathcal{K}_j$ be the set of trigger words for the event mentions of the subtype $s_j \in \mathcal{S}$. $\mathcal{K}$ is collected from training set of ACE 2005.

To generate the training and test/development datasets, we first split the documents in ACE 2005 into three parts D_{train}, D_{test} and D_{dev} following the previous work on ED (Li et al., 2013). They would contain event mentions for all the possible event types and subtypes in $\mathcal{T}$ and $\mathcal{S}$. Assume that we want to extend the system to a new event type $t_{target} \in \mathcal{T}$, we need a train set without t_{target}. So, we remove every event mention whose subtype belongs to $\mathcal{C}_{target}$ from D_{train}. Whereas, samples with subtypes in $\mathcal{C}_{target} \cup \{Other\}$ are kept in D_{test} and D_{dev}. The results of this removal process are called as D'_{train}, D'_{test} and D'_{dev} (from D_{train}, D_{test} and D_{dev}, respectively). They will be used to generate the actual training/test/development datasets for LFK, respectively.

Specifically, for each of these datasets (i.e., D'_{train}, D'_{test} and D'_{dev}), the goal is to produce the positive and negative examples in corresponding LFK datasets. Algorithm 1 shows the pseudocode to generate the training dataset for LFK from D'_{train}. The same algorithm can be applied for the test and development dastasets of LFK, but replace D'_{train} with D'_{test} and D'_{dev} respectively in line 2, and replace $\mathcal{S} \setminus \mathcal{C}_{target}$ with $\mathcal{C}_{target}$ in line 10.

Since the number of positive examples in D_{test} set is small, we choose two event types (i.e., *Conflict* and *Life*) that have the largest numbers of positive examples in D_{test} as the target types. Applying the data generation procedure above, we generate a dataset in LFK for each of these target types.

Algorithm 1 Training dataset generation for LFK

```
1:  D+_train, D-_train ← ∅, ∅   ▷ Positive and negative example sets
2:  for (X, x_a, s_j) ∈ D'_train do   ▷ where X : a sentence, x_a ∈ X : the anchor word, s_j ∈ S : the corresponding subtype
3:      if s_j ≠ "Other" then
4:          for u = 1..5 do
5:              K^u_j ← A subset of K_j \ {x_a}: |K^u_j| = 4
6:              D+_train ← D+_train ∪ {(X, x_a, K^u_j, 1)}
7:          end for
8:      else   ▷ s = "Other"
9:          s_v ← Some subtype in S \ C_target
10:         K ← A subset of K_v: |K| = 4
11:         D-_train ← D-_train ∪ {(X, x_a, K, 0)}
12:     end if
13: end for
14: return D+_train and D-_train
```

leave the other models for future research.

Encoding Layer: To prepare the sentence S and the anchor x_a for the models, we first convert each word $x_i \in S$ into a concatenated vector $h^0_i = [p_i, q_i]$, in which $p_i \in \mathbb{R}^u$ is the position embedding vector and $q_i \in \mathbb{R}^d$ is the word embedding of x_i. We follow the settings for p_i and q_i described in (Nguyen and Grishman, 2015). This step transforms S into a sequence of vector $H^0 = (h^0_1, h^0_2, \ldots, h^0_n)$.

Convolution Layers: Following (Chen et al., 2015; Nguyen and Grishman, 2015), we apply a convolutional layers with multiple window sizes for the filters W over H_0, resulting in a sequence of hidden vectors $H^1 = (h^1_1, h^1_2, \ldots, h^1_n)$. Note that we pad H^0 with zero vectors to ensure that H^1 still has n vectors. We can essentially run m convolutional layers in this way that would lead to m sequences of hidden vectors $H^1, H^2, \ldots, H^m$.

Keyword Representation: We generate the representation vector V_K for the keyword set K by taking the average of the embeddings of its words.

Given the keyword vectors V_K and the hidden vector sequences from CNNs for S (i.e., $H^1, H^2, \ldots, H^m$), the goal is to produce the final representation $R = G(V_K, H^1, H^2, \ldots, H^m)$, serving as the features to predict the matching between (S, x_a) and K (i.e., R would be fed into a feed-forward neural network with a softmax layer

4 Model

This section first presents the typical deep learning models in the traditional ED formulation adapted to LFK. We then introduce a novel attention mechanism to improve such models for LFK.

4.1 Baselines

As CNNs have been applied to the traditional formulation of ED since the early day (Chen et al., 2015; Nguyen and Grishman, 2015, 2016), we focus on the CNN-based model in this work and in the end to perform classification). There are two immediate baselines to obtain R adapted from the models for traditional ED:

(i) *Concat*: In this method, we apply the usual max-pooling operation over the hidden vectors for the last CNN layer H^m whose result is concatenated with V_K to produce R (Nguyen and Grishman, 2015; Chen et al., 2015).

(ii) *Attention*: This method applies the popular attention method to aggregate the hidden vectors in H^m using V_K as the query (Bahdanau et al., 2015). The formulas for R are shown below:

$$
\begin{aligned}
u_i &= \sigma(W_u h_i^m + b_u) \\
c &= \sigma(W_c[V_K, h_a^m] + b_c) \\
\alpha_i &= \frac{\exp(c^\top u_i)}{\sum_j \exp(c^\top u_j)} \\
R &= \sum_i \alpha_i h_i^m
\end{aligned}
$$

4.2 Conditional Feature-wise Attention

The interaction between the keywords and hidden vectors in the baselines is only done in the last layer, letting the intermediate CNN layers to decide the computation themselves without considering the information from the keywords.

To overcome this limitation, we propose to inject supervision signals for each CNN layer in the modeling process. In particular, given the sequence of hidden vectors $H^i = (h_1^i, h_2^i, \ldots, h_n^i)$ obtained by the i-th CNN layer, instead of directly sending H^i to the next layer, we use V_K to generate the representation vectors γ_i and β_i, aiming to reveal the underlying information/constraints from the keywords that the i-th CNN layer should reason about. Such representation vectors condition and bias the hidden vectors in H^i toward the keywords based on the feature-wise affine transformation (Perez et al., 2018). The conditioned hidden vectors from this process (called $\bar{H}^i = (\bar{h}_1^i, \bar{h}_2^i, \ldots, \bar{h}_n^i)$) would be sent to the next CNN layer where the conditional process guided by the keywords continues:

$$
\begin{aligned}
\gamma_i &= \sigma(W_\gamma^i V_K + b_\gamma^i) \\
\beta_i &= \sigma(W_\beta^i V_K + b_\beta^i) \\
\bar{h}_j^i &= \gamma_i * h_j^i + \beta_i
\end{aligned}
$$

where σ is a non-linear function while W_γ^i, b_γ^i, W_β^i and b_β^i are model parameters.

We call the operation described in this section the Conditional Feature-wise Attention (CFA). The application of CFA into the two baselines **Concat** and **Attention** leads to two new methods **Concat-CFA** and **Attention-CFA** respectively.

5 Experiments

We use the datasets generated in Section 3.2 to evaluate the models in this section. Table 1 shows the staticstics of our generated datasets. This dataset will be publicly available to the community.

	Label	Train	Dev	Test
Conflict	+1	14,749	929	509
	-1	177,421	13,130	13,576
Life	+1	17,434	354	154
	-1	177,421	13,130	13,576

Table 1: Numbers of the positive and negative samples of the LFK datasets.

5.1 Parameters

We examine four deep learning models: baselines (i.e., *Concat* and *Attention*) and the proposed models (i.e., *Concat-CFA* and *Attention-CFA*). Following (Nguyen and Grishman, 2015), we employ the `word2vec` word embeddings from (Mikolov et al., 2013) with 300 dimensions for the models in this work. The other parameters for the deep learning models in this work are tuned on the development datasets. In particular, we employ multiple window sizes (i.e., 2, 3, 4 and 5) in the CNN layers, each has 100 filters. We use Adadelta as the optimizer with the learning rate set to 1.0. We apply a dropout with a rate of 0.5 to the final representation vector R. Finally, we optimize the number of CNN layers for each deep learning model.

In addition, we investigate the typical feature-based models with the MaxEnt classifier in the traditional ED formulation for LFK to constitute the baselines for future research. In particular, we examine four feature-based models for ED in LFK:

- **Feature** combines the state-of-the-art feature set for ED designed in (Li et al., 2013) for the input context (S, x_a) with the words in the keyword set K to form its features

- **Word2Vec** utilizes the keyword representation V_K and the average of the embedding of the words in the window size of 5 for x_a in S as the features

Model	Conflict			Life		
	P	R	F1	P	R	F1
Feature	21.7	9.8	13.5	14.4	25.8	18.5
Word2Vec	20.6	72.4	32.1	4.4	61.9	8.2
Feature + Word2vec	27.8	20.2	23.4	15.7	31.6	21.0
Seed	11.9	36.1	17.9	9.5	71.0	16.7
Concat	20.5	57.8	30.0 (4)	10.9	48.3	17.7 (2)
Attention	21.5	59.1	31.4 (4)	12.8	45.0	19.1 (2)
Concat-CFA	25.1	57.1	33.8 (4)	10.6	43.6	16.9 (1)
Attention-CFA	22.5	74.2	**34.1** (1)	18.5	38.7	**25.0** (4)

Table 2: Model performance. The numbers in the brackets indicate the optimized numbers of CNN layers.

- **Feature + word2vec** uses the aggregated features from above models

- **Seed** employs the model with semantic features in (Bronstein et al., 2015).

5.2 Evaluation

Table 2 presents the performance of the models on the test performance for different datasets (i.e., with *Conflict* and *Life* as the target type). Among the features in the feature-based models, the embedding features in *Word2Vec* are very helpful for ED in LFK as the models with these features achieve the best performance (i.e., *Word2Vec* for *Conflict* and *Feature+Word2Vec* for *Life*). Among the deep learning models, the CFA-based models (i.e., *Concat-CFA* and *Attention-CFA*) are significantly better than their corresponding baseline models (i.e., *Concat* and *Attention*) over both *Conflict* and *Life* with *Attention*. This confirms the benefits of CFA for ED in LFK.

Comparing the deep learning and the feature-based models, it is interesting that the feature-based models with average word embedding features can perform better than the deep learning baseline models (i.e., *Concat* and *Attention*) for *Conflict*. However, when the deep learning models are integrated with both attention and CFA (i.e., *Attention-CFA*), it achieves the best performance over both datasets. This helps to testify to the advantage of deep learning and CFA for ED in the new type extension setting with LFK.

Finally, although the models can extract event mentions of the new types, the performance is still limited in general, illustrating the challenge of ED in this setting and leaving many rooms for future research (especially with deep learning) to improve the performance. We hope that the setting in this work presents a new way to evaluate the effectiveness of the ED models.

6 Conclusion

We investigate a new formulation for event detection task that enables the operation of the models to new event types, featuring the use of keywords to specify the event types on the fly for the models. A novel feature-wise attention technique is presented for the CNN models for ED in this formulation. Several models are evaluated to serve as the baselines for future research on this problem.

References

David Ahn. 2006. The stages of event extraction. In *Proceedings of the Workshop on Annotating and Reasoning about Time and Events*, pages 1–8.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *ICLR*.

Ofer Bronstein, Ido Dagan, Qi Li, Heng Ji, and Anette Frank. 2015. Seed-based event trigger labeling: How far can event descriptions get us? In *ACL-IJCNLP (Volume 2: Short Papers)*, volume 2, pages 372–376.

Yubo Chen, Shulin Liu, Xiang Zhang, Kang Liu, and Jun Zhao. 2017. Automatically labeled data generation for large scale event extraction. In *ACL*.

Yubo Chen, Liheng Xu, Kang Liu, Daojian Zeng, and Jun Zhao. 2015. Event extraction via dynamic multi-pooling convolutional neural networks. In *ACL-IJCNLP (Volume 1: Long Papers)*, volume 1, pages 167–176.

Yu Hong, Jianfeng Zhang, Bin Ma, Jianmin Yao, Guodong Zhou, and Qiaoming Zhu. 2011. Using cross-entity inference to improve event extraction. In *ACL*.

Lifu Huang, Heng Ji, Kyunghyun Cho, and Clare R Voss. 2018. Zero-shot transfer learning for event extraction. In *ACL*, pages 2160–2170.

Heng Ji and Ralph Grishman. 2008. Refining event extraction through cross-document inference. In *ACL*.

Qi Li, Heng Ji, Yu Hong, and Sujian Li. 2014. Constructing information networks using one single model. In *EMNLP*.

Qi Li, Heng Ji, and Liang Huang. 2013. Joint event extraction via structured prediction with global features. In *ACL*.

Xiang Li, Thien Huu Nguyen, Kai Cao, and Ralph Grishman. 2015. Improving event detection with abstract meaning representation. In *Proceedings of the First Workshop on Computing News Storylines*.

Shasha Liao and Ralph Grishman. 2010. Using document level cross-event inference to improve event extraction. In *ACL*.

Jian Liu, Yubo Chen, Kang Liu, and Jun Zhao. 2018. Event detection via gated multilingual attention mechanism. In *AAAI*.

Shulin Liu, Yubo Chen, Kang Liu, and Jun Zhao. 2017. Exploiting argument information to improve event detection via supervised attention mechanisms. In *ACL*.

Weiyi Lu and Thien Huu Nguyen. 2018. Similar but not the same: Word sense disambiguation improves event detection via neural representation matching. In *EMNLP*.

David McClosky, Mihai Surdeanu, and Christopher Manning. 2011. Event extraction as dependency parsing. In *BioNLP Shared Task Workshop*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In *NIPS*.

Makoto Miwa, Paul Thompson, Ioannis Korkontzelos, and Sophia Ananiadou. 2014. Comparable study of event extraction in newswire and biomedical domains. In *COLING*.

Minh Nguyen and Thien Huu Nguyen. 2018b. Who is killed by police: Introducing supervised attention for hierarchical lstms. In *COLING*.

Thien Nguyen and Ralph Grishman. 2018. Graph convolutional networks with argument-aware pooling for event detection. In *AAAI*.

Thien Huu Nguyen, , Adam Meyers, and Ralph Grishman. 2016a. New york university 2016 system for kbp event nugget: A deep learning approach. In *TAC*.

Thien Huu Nguyen, Kyunghyun Cho, and Ralph Grishman. 2016b. Joint event extraction via recurrent neural networks. In *NAACL*.

Thien Huu Nguyen, Lisheng Fu, Kyunghyun Cho, and Ralph Grishman. 2016c. A two-stage approach for extending event detection to new types via neural networks. In *Proceedings of the 1st ACL Workshop on Representation Learning for NLP (RepL4NLP)*.

Thien Huu Nguyen and Ralph Grishman. 2015. Event detection and domain adaptation with convolutional neural networks. In *ACL-IJCNLP*.

Thien Huu Nguyen and Ralph Grishman. 2016. Modeling skip-grams for event detection with convolutional neural networks. In *EMNLP*.

Trung Minh Nguyen and Thien Huu Nguyen. 2019. One for all: Neural joint modeling of entities and events. In *AAAI*.

Haoruo Peng, Yangqiu Song, and Dan Roth. 2016. Event detection and co-reference with minimal supervision. In *EMNLP*, pages 392–402.

Ethan Perez, Florian Strub, Harm De Vries, Vincent Dumoulin, and Aaron Courville. 2018. Film: Visual reasoning with a general conditioning layer. In *AAAI*.

Sebastian Riedel, Hong-Woo Chun, Toshihisa Takagi, and Jun'ichi Tsujii. 2009. A markov logic approach to bio-molecular event extraction. In *BioNLP 2009 Workshop*.

Sebastian Riedel and Andrew McCallum. 2011a. Fast and robust joint models for biomedical event extraction. In *EMNLP*.

Sebastian Riedel and Andrew McCallum. 2011b. Robust biomedical event extraction with dual decomposition and minimal domain adaptation. In *BioNLP Shared Task 2011 Workshop*.

Lei Sha, Feng Qian, Baobao Chang, and Zhifang Sui. 2018. Jointly extracting event triggers and arguments by dependency-bridge rnn and tensor-based argument interaction. In *AAAI*.