Knowledge-Enhanced Self-Supervised Prototypical Network for Few-Shot Event Detection

Kailin Zhao, Xiaolong Jin*, Long Bai, Jiafeng Guo, Xueqi Cheng
CAS Key Laboratory of Network Data Science and Technology, Institute of Computing Technology, Chinese Academy of Sciences;
School of Computer Science and Technology, University of Chinese Academy of Sciences
{zhaokailin17z, jinxiaolong, bailong18b}@ict.ac.cn
{guojiafeng, cxq}@ict.ac.cn

Abstract

Prototypical network based joint methods have attracted much attention in few-shot event detection, which carry out event detection in a unified sequence tagging framework. However, these methods suffer from the inaccurate prototype representation problem, due to two main reasons: the number of instances for calculating prototypes is limited; And, they do not well capture the relationships among event prototypes. To deal with this problem, we propose a Knowledge-Enhanced self-supervised Prototypical Network, called KE-PN, for few-shot event detection. KE-PN adopts hybrid rules, which can automatically align event types to an external knowledge base, i.e., FrameNet, to obtain more instances. It proposes a self-supervised learning method to filter out noisy data from enhanced instances. KE-PN is further equipped with an auxiliary event type relationship classification module, which injects the relationship information into representations of event prototypes. Extensive experiments on three benchmark datasets, i.e., FewEvent, MAVEN, and ACE2005 demonstrate the state-of-the-art performance of KE-PN.

1 Introduction

Event detection is fundamental to information extraction, which consists of two sub-processes, i.e., trigger word identification and event classification. The former extracts the triggers from a piece of text describing events, while the latter classifies them into different event types. For example , in "*Our college is to make arrangements for the meeting*", the trigger words are *"make arrangements*", indicating an ***Arranging*** event. Event detection benefits many downstream applications, e.g., question answering and information retrieval.

Typical methods for event detection (Chen et al., 2015; Nguyen and Grishman, 2018; Liu et al., 2019) have been heavily dependent on a large quantity of labeled data. However, in many real-world scenarios, labeled data are often inadequate, which limits the performance of existing methods. Therefore, researchers have shown increasing interests in event detection with only a few labeled instances, which is thus called Few-Shot Event Detection (FSED).

There are two kinds of approaches in FSED, namely, pipeline ones (Lai et al., 2020; Deng et al., 2020, 2021) and joint ones (Lai et al., 2021; Cong et al., 2021; Chen et al., 2021). The former adopt a two-stage (i.e., identification and classification) process, while the latter regard the two sub-processes as a joint one. Since the joint approaches alleviate the error propagation problem appeared in pipeline ones, they have become the mainstream ones. In the joint approaches, FSED is formulated as a sequence tagging task, where each word in a sequence is assigned a label. The label consists of two parts: the position part and the type part (Fritzler et al., 2019). There are three types of labels for the position part, i.e., B, I and O, where B and I indicate the beginning and inside positions of the corresponding words in the event triggers, respectively, and O refers to other words (i.e., non-trigger ones). The type part indicates the event type of the instance. Moreover, this kind of approaches usually adopts Prototypical Network (PN) (Snell et al., 2017) as their classifier, whose main idea is to learn a metric space in which classification of an instance can be performed by measuring its distance to different prototypes. An example of "BIO"-based sequence tagging PN for FSED is shown in Figure 1. Furthermore, the ***B-Arranging*** and ***I-Arranging*** prototypes compose the ***Arranging*** event prototype.

A challenging problem of these PN-based FSED approaches is how to obtain accurate prototype representations, because of two main reasons: First, the number of instances for calculating event proto-

*Corresponding author.

Findings of the Association for Computational Linguistics: EMNLP 2022, pages 6266–6275
December 7-11, 2022 ©2022 Association for Computational Linguistics

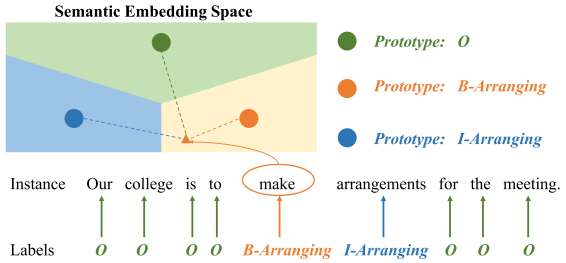

Figure 1: An example of sequence tagging based PN for FSED.

types is limited in the few-shot scenarios. In FSED, the trigger words account for only a small proportion of all tokens in a word sequence, which makes the tokens with labels B and I even fewer. Hence, the prototype representations for labels B and I become less accurate. Second, existing approaches usually assume by default that event prototypes are independent. Therefore, they fail to capture the relationships (i.e., the parent-child relationship and the sibling relationship) among these prototypes.

To solve the above problem, we propose a novel **K**nowledge-**E**nhanced self-supervised **P**rototypical **N**etwork, called KE-PN, for FSED. To obtain more accurate prototype representations, KE-PN adopts a novel knowledge enhancement method which introduces knowledge from an external knowledge base, i.e., FrameNet (Baker et al., 1998), when computing the prototypes. Unlike recent approaches relying on a mixture of string matching and human annotation (Shen et al., 2021), KE-PN applies hybrid rules which align event types to the frames in FrameNet via a completely automatic manner. Then, KE-PN replaces the triggers of the support instances with the LexiUnits in the aligned frames to form new instances. To reduce the noise brought by the above method, KE-PN adopts a self-supervised learning method to filter out noise from the enhanced support set. Moreover, so as to inject relationship information into prototype representations, KE-PN is equipped with an auxiliary event type relationship classification module.

In summary, the main contributions of this paper are three-fold.

1) We propose a novel knowledge-enhanced self-supervised learning method to well calculate the representations of event prototypes for the prototypical network, by introducing the knowledge from an external knowledge base, i.e., FrameNet.

2) We adopt event type relationship classification as an auxiliary module, to inject relationship information into prototype representations.

3) Extensive experiments on three benchmark datasets, i.e., FewEvent, MAVEN and ACE2005 demonstrate the state-of-the-art performance of KE-PN.

2 Related Works

2.1 Few-Shot Event Detection

As aforesaid, there are two kinds of approaches in FSED, i.e., pipeline and joint approaches. Under the pipeline framework, Lai et al. (2020) were the first to apply few-shot learning into event detection, and thus introduced two regularization matching losses to improve the performance of the models. Then, Deng et al. (2020) proposed a standard FSED dataset, called FewEvent, and designed DMB-PN, a dynamic memory based network. To introduce the external knowledge, Shen et al. (2021) presented AKE-BML based on the Bayesian method, which adopts string matching and human annotation to align the event types to FrameNet. However, these pipeline approaches follow the identification-then-classification process and thus suffer from the error propagation problem. Due to this reason, joint approaches in FSED have attracted much attention. Cong et al. (2021) firstly solved FSED with two sub-processes in a unified manner and proposed PA-CRF based on the sequence tagging method. Later on, in order to solve the trigger curse problem in FSED which means overfitting the trigger will harm the generalization ability, whilst underfitting it will hurt the detection performance, Chen et al. (2021) proposed a structural causal model. These joint approaches usually employ PN (Snell et al., 2017) as their classifier and have achieved promising performance. However, they still suffer from inaccurate prototype representations. To overcome this challenge, we propose KE-PN to enhance the prototype representations and thus obtain more accurate prototypes.

2.2 Prototypical Network

The original PN is to learn a metric space in which classification of an instance can be performed by computing its distances to different prototypes (Snell et al., 2017). The prototype $\boldsymbol{c}_i$ of the i-th class in the support set is a representative vector. It is calculated upon averaging the vectors of the support instances as $\boldsymbol{c}_i = \frac{1}{K}\sum_{j=1}^{K} \boldsymbol{x}_i^j$, where $\boldsymbol{x}_i^j$ indicates the representation of the j-th instance of the i-th class. Then, by calculating the distance be-

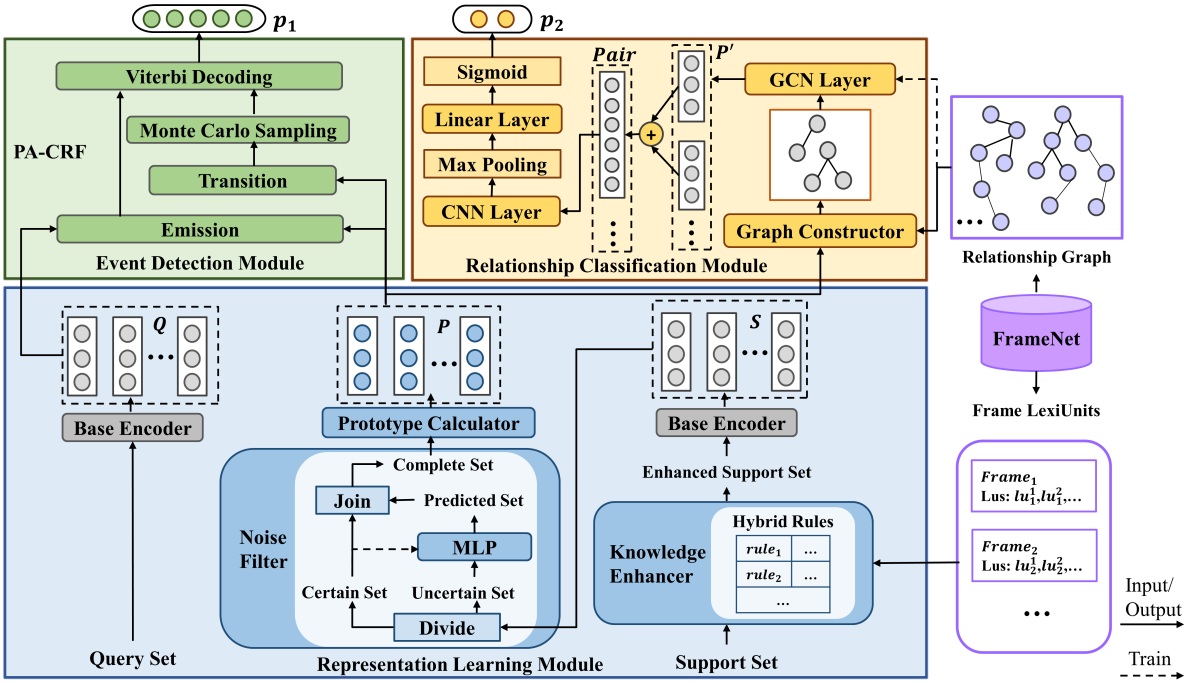

Figure 2: The diagram of the KE-PN model.

tween the representation vector of a query instance and the prototype vectors, we can obtain a distance-based distribution over the possible classes in the current episode,

$$p(y = t_i|x) = \frac{\exp(-d(f(x), \boldsymbol{c}_i))}{\sum_{j=1}^{N} \exp(-d(f(x), \boldsymbol{c}_j))}, \quad (1)$$

where $d(.,.)$ is a distance function (e.g., Euclidean distance). For sequence tagging based PN, the prototype $\boldsymbol{c}_i$ is calculated upon averaging the representations of tokens with the i-th label, which can be *B-EventType*, *I-EventType* or O. The prototypes of *B-EventType* and *I-EventType* compose the corresponding event prototype.

3 Notations

In FSED, two datasets are given: D_{train} and D_{test}, which have disjoint event type sets. Each dataset contains several tasks and each task consists of a support set and a query set, which is formulated in the N-way K-shot paradigm. Given the support set $S = \{(x_i, l_i)\}_{i=1}^{N \times K}$ which has N classes and each class has K labeled instances, FSED aims to predict the labels of tokens in the query set $Q = \{q_i\}_{i=1}^{U}$. In the support set S, $x_i = \{w_i^1, w_i^2, ..., w_i^n\}$ denotes a n-word sequence, and $l_i = \{l_i^1, l_i^2, ..., l_i^n\}$ denotes its label sequence. The query set Q contains U instances, where q_i refers to a sequence of unlabeled tokens. Since FSED is formulated as a sequence tagging process, the total number of prototype labels is $2N + 1$ (N for *B-EventType*, another N for *I-EventType*, and 1 for label O).

4 The KE-PN Method

The KE-PN method consists of three modules, i.e., representation learning, event detection and event type relationship classification, as shown in Figure 2.

Representation Learning. This module aims to obtain the representations of event prototypes and query instances. To obtain accurate prototype representations, a knowledge enhancement self-supervised learning method is applied.

Event Detection. This module takes the representations of prototypes and query instances as its input, and predicts the labels of tokens in the query set. We adopt PA-CRF, which is the state-of-the-art FSED model, as the event detection method.

Event Type Relationship Classification. This module takes the prototype representations as its input, and predicts whether two prototypes have concerned relationships. This module injects relationship information into prototype representations by working as an auxiliary module.

No.	Alignment Conditions
1	t_i.eql(f)
2	For any ten in $p(t_i)$: ten.eql(f)
3	For any nou in $n(t_i)$: nou.eql(f)
4	For any syn in $s(t_i)$: syn.eql(f)
5	t_i.con(f) or f.con(t_i)
6	For any ten in $p(t_i)$: ten.con(f) or f.con(ten)
7	For any nou in $n(t_i)$: nou.con(f) or f.con(nou)
8	For any syn in $s(t_i)$: syn.con(f) or f.con(syn)

Table 1: The conditions for aligning event types to frames.

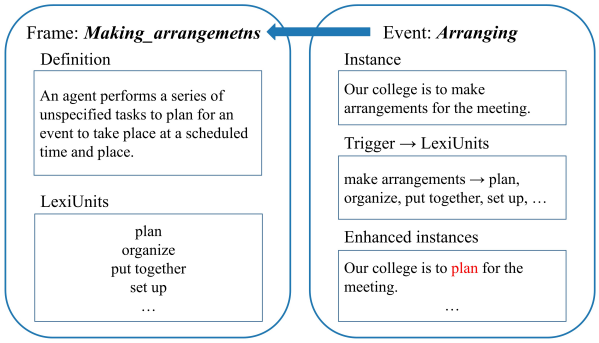

Figure 3: An example aligning an ***Arranging*** event to the ***Making_arrangements*** frame in FrameNet.

4.1 Representation Learning

This module includes four components, i.e., knowledge enhancer, base encoder, noise filter and prototype calculator. Given the support set S, the knowledge enhancer produces the enhanced support set upon introducing external knowledge. Then, by inputting the enhanced support set, the base encoder maps these instances into a semantic embedding space. After that, the noise filter removes noise from the enhanced support set. Finally, the prototype calculator computes the prototypes upon averaging the instance vectors obtained from the last step.

4.1.1 Knowledge Enhancer

The knowledge enhancer presents a novel knowledge enhancement method for FSED, which is based on hybrid rules. Previous works have aligned event types to external knowledge bases (Shen et al., 2021). However, they use manpower, which is time-consuming and expert-driven. For this reason, we design hybrid rules which align the event types to the frames in FrameNet via a completely automatic manner, as shown in Table 1. Let t_i denote the i-th event type, f denote the candidate frame and F_i denote the corresponding frame set. We adopt WordNet (Miller, 1995) to get the nouns and synonyms of a word, where $p(\cdot)$ represents the past tense and the present progressive of a word, $n(\cdot)$ denotes the nouns of a word, and $s(\cdot)$ indicates the synonyms of a word. And, a.eql(b) means string a is the same as string b and a.con(b) indicates b is a substring of a. If t_i and f satisfy any of these rules, f should be put into F_i.

Then, we replace the triggers in the support instances with the LexiUnits of the aligned frames in FrameNet to obtain the enhanced instances, as shown in Figure 3. Let M denote the max number of enhanced instances for each class. To ensure the same number of instances for all classes, we add zero vectors for the classes whose number of enhanced instances is less than M. Therefore, the original instances and the enhanced ones compose the final enhanced support set S' with $N \times (K+M)$ instances.

4.1.2 Base Encoder

The base encoder aims to map the instances in the enhanced support set S' and query set Q into the embedding space to express their semantic meanings. Given the input $x_i = \{w_i^1, w_i^2, ..., w_i^n\}$, BERT (Kenton and Toutanova, 2019) is employed to get the embedding representations of x_i as follows,

$$\boldsymbol{x}_i = \{\boldsymbol{w}_i^1, \boldsymbol{w}_i^2, ..., \boldsymbol{w}_i^n\} = BERT(x_i), \quad (2)$$

where $\boldsymbol{w}_i^j$ denotes the representation of token w_i^j, which is of B dimension. Thus, the embedding set $\boldsymbol{S}$ of S' can be formulated as

$$\boldsymbol{S} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_{N\times(K+M)}\}. \quad (3)$$

Similarly, the embedding set $\boldsymbol{Q}$ of Q is formulated as

$$\boldsymbol{Q} = \{\boldsymbol{q}_1, \boldsymbol{q}_2, ..., \boldsymbol{q}_U\}, \quad (4)$$

where $\boldsymbol{q}_i$ denotes the embedding representation of q_i by

$$\boldsymbol{q}_i = BERT(q_i). \quad (5)$$

4.1.3 Noise Filter

Given $\boldsymbol{S}$ from the base encoder, we have statistically analyzed how many wrong frames are aligned by the knowledge enhancer. The analysis results are shown in Table 2, where we can see that the

Dataset	#Class	#Aligned Frames	#Wrong Frames	#Aligned Frames per Class	#Wrong Frames per Class
FewEvent	100	317	244	3.17	2.44
ACE2005	33	91	70	2.76	2.12

Table 2: The statistic of aligned frames and wrong frames in FewEvent and ACE2005.

wrong frames have accounted for most on the FewEvent and ACE2005 datasets. Note that the statistics of MAVEN is not presented, as it has golden alignment to FrameNet. Especially, the incorrect instances obtained by the knowledge enhancer from the wrong frames are called noise.

Due to the different reliability of hybrid rules, the enhanced instances can be divided into certain instances and uncertain instances. Among them, the uncertain ones contain more noise. To filter out the noise from the uncertain ones, we propose a self-supervised learning method, which utilizes the certain instances as the training set.

Specifically, we divide the given enhanced support set $\boldsymbol{S}$ into the certain set cer_i and the uncertain set unc_i for event type t_i. The positive instances of cer_i include the original support instances of t_i and the instances obtained via the hybrid rules of Nos. 1, 2, 3 and 4. The negative instances of cer_i include the certain set of other event types in the given support set. unc_i is composed of the instances obtained by the hybrid rules of Nos. 5, 6, 7 and 8. Then, cer_i is exploited to train a binary classifier, for which a Multi-Layer Perception (MLP) (Murtagh, 1991) is adopted. Then, we obtain the predicted positive instances in unc_i upon inputting unc_i into the trained MLP. Finally, the original positive instances in cer_i and the predicted positive instances in unc_i compose the complete support set CS.

4.1.4 Prototype Calculator

The prototype calculator is to obtain the prototype representation upon averaging the token vectors for each label. The prototype representation $\boldsymbol{c}_i$ for the i-th label ($i \in [1, 2N+1]$) is calculated by

$$\boldsymbol{c}_i = \frac{1}{|W(CS, l_i)|} \sum_{w \in W(CS,l_i)} \boldsymbol{w}, \tag{6}$$

where $W(CS, l_i)$ indicates the token set with label l_i in CS and $\boldsymbol{w}$ refers to the representation of token w. In addition, $|\cdot|$ denotes the number of elements in the set. Finally, the embedding set $\boldsymbol{P}$ of all prototypes is presented as

$$\boldsymbol{P} = \{\boldsymbol{c}_i\}_{i=1}^{2N+1}. \tag{7}$$

4.2 Event Detection

In this module, we adopt PA-CRF for event detection, which mainly consists of three sub-modules, i.e., emission module, transition module and decoding module.

The emission module aims to calculate the emission score for each token in the query set Q. The emission scores are obtained upon calculating the similarities between the presentations of the query token and the prototypes. In practice, the dot product operation is chosen to measure the similarity.

The transition module is to generate the distributional parameters (i.e., mean and variance) of transition scores based on the label prototypes.

The decoding module derives the probability for a specific label sequence of the query tokens according to the emission scores and approximated Gaussian distributions of transition scores. The Monte Carlo sampling technique (Gordon et al., 2019) is employed to approximate the integral. In the inference phase, PA-CRF adopts the Viterbi algorithm (Forney, 1973) to decode the probability distribution of the best-predicted label sequence $\boldsymbol{p}_1$ to different label sequences for the query tokens. The event detection process can be simplified as

$$\boldsymbol{p}_1 = PA - CRF(\boldsymbol{P}, \boldsymbol{Q}). \tag{8}$$

Then, the loss l_1 of this module is obtained upon the cross entropy loss function $L(.,.)$ as

$$l_1 = L(\boldsymbol{p}_1, \boldsymbol{y}_1), \tag{9}$$

where $\boldsymbol{y}_1$ denotes the ground truth distribution of the query tokens to different label sequences.

4.3 Event Type Relationship Classification

So as to inject relationship information into prototype representations, we exploit event type relationship classification as an auxiliary module for FSED. In this module, the relationships which we concerned are parent-child and sibling relationships. Therefore, two prototypes are related, if they or their corresponding frames have the above two relationships.

First of all, a Graph Conventional Network (GCN) (Scarselli et al., 2009) is pre-trained on the

graph which contains these two relationships of FrameNet. The representation of each frame is obtained upon inputting its definition into the base encoder.

Given the embedding set $\boldsymbol{P}$ of prototypes and the relationship graph as input, we construct the adjacency matrix $\boldsymbol{A}$ for prototypes with label B. If two prototypes are related, their adjacency weight is set to 1; Otherwise, the weight is 0. Then, we adopt the pre-trained GCN as the encoder, which takes $\boldsymbol{P}$ and their adjacency matrix $\boldsymbol{A}$ as its input, to obtain the updated prototype representations $\boldsymbol{P}^{'}$ as

$$\boldsymbol{P}^{'} = GCN(\boldsymbol{P}, \boldsymbol{A}). \tag{10}$$

Then, we concatenate any two prototypes by

$$\boldsymbol{Pair}_{m,n} = concat(\boldsymbol{P}^{'}_{m}, \boldsymbol{P}^{'}_{n}), \tag{11}$$

where $m \neq n$ and $m, n \in [1, N]$.

The Conventional Neural Network (CNN) decoder is employed as the relationship classifier, to predict whether two prototypes are related. It slides a conventional kernel, whose window size is k, over the concatenated embeddings to get the output hidden embeddings,

$$\boldsymbol{h}_{m,n} = \mathrm{Con}\left(\boldsymbol{Pair}_{m,n}\right), \tag{12}$$

where $\mathrm{Con}\left(\cdot\right)$ is a conventional operation.

A max pooling operation is then applied over these hidden embeddings to output the final embedding $\boldsymbol{h}^{'}_{m,n}$ as follows:

$$\boldsymbol{h}^{'}_{m,n} = max\left\{[\boldsymbol{h}_{m,n}]_1, ..., [\boldsymbol{h}_{m,n}]_b, ...\right\}, \tag{13}$$

where $[\cdot]_b$ is the b-th value of a vector ($b \in [1, B \times 2]$).

Then, we employ Sigmoid as the activation function and thus obtain the probability distribution $\boldsymbol{p}_2$ of whether two prototypes are related. The loss l_2 of the relationship classification module is calculated by the cross entropy loss function $L(.,.)$ as

$$l_2 = L(\boldsymbol{p}_2, \boldsymbol{y}_2), \tag{14}$$

where $\boldsymbol{y}_2$ denotes the ground truth distribution to different relationships between two prototypes. For the MAVEN dataset, its classification labels include parent-child and non-parent-child relationships. Moreover, the relationship labels are sibling and non-sibling for FewEvent and ACE2005 datasets.

Dataset	#Class	#Train	#Dev	#Test
FewEvent	100	80	10	10
ACE2005	33	13	10	10
MAVEN	100	64	16	20

Table 3: The statistic of classes split in the three benchmark datasets.

Finally, the overall loss l is obtained by the sum of l_1 and l_2 as

$$l = l_1 + l_2. \tag{15}$$

The parameters of KE-PN is updated by minimizing the loss l through applying gradient-based optimization.

5 Experiments

5.1 Datasets and Evaluation Metrics

As aforesaid, we conduct experiments on the three FSED benchmark datasets, i.e., FewEvent (Deng et al., 2020), MAVEN (Wang et al., 2020), and ACE2005 (Doddington et al., 2004). For FewEvent, we adopt the version split by (Cong et al., 2021), which contains 80, 10 and 10 event types for training, validation and test, respectively. To match the number of classes for the standard few-shot dataset, i.e., FewEvent, we adopt 100 classes in MAVEN which have more than 200 instances and randomly divide them into subsets with 64, 16 and 20 classes for training, validation and test, respectively. For ACE2005 which have 33 classes, we randomly partition it into 13, 10 and 10 classes for training, validation and test, respectively. The statistics of the three datasets are shown in Table 3.

We set up four configurations, namely, 5-way 1-shot, 5-way 5-shot, 10-way 1-shot and 10-way 5-shot, for each FSED task on three datasets. In addition, the same as the previous works (Chen et al., 2015; Liu et al., 2018; Cui et al., 2020), we adopt the standard micro F1 score as the evaluation metric and report the averages and standard deviations upon 5 randomly initialized runs.

5.2 Implementation Details and Parameter Setting

The parameter setting is as follows. For the representation learning module, the number of enhanced instances M is set to 25, for the balance of performance and resource. BERT-base-uncased (Kenton and Toutanova, 2019) is employed as the base encoder, whose input sentence is of 128 max length and the hidden size B is 768. For the noise filter, we

Dataset: FewEvent				
Method	5-way 1-shot	5-way 5-shot	10-way 1-shot	10-way 5-shot
Match	24.70±2.45	39.93±0.67	18.35±1.34	30.88±1.08
Proto	22.72±0.24	50.11±0.77	17.49±0.02	43.51±1.16
Proto-dot	49.10±0.01	58.82±0.88	44.88±0.01	55.04±1.62
Relation	11.37±0.02	28.91±1.13	7.15±0.01	18.49±1.25
PA-CRF	48.63±0.12	62.25±1.42	43.91±0.07	58.48±0.68
KE-PN	**74.85±0.04**	**78.19±0.05**	**74.29±0.02**	**78.81±0.03**
Dataset: MAVEN				
Method	5-way 1-shot	5-way 5-shot	10-way 1-shot	10-way 5-shot
Match	10.43±0.01	19.30±0.04	8.41±0.01	12.55±0.01
Proto	11.50±0.02	33.63±0.05	9.36±0.01	24.73±0.01
Proto-dot	43.29±0.01	61.92±0.01	30.96±0.03	56.91±0.01
Relation	0.04±0.01	0.48±0.01	0.01±0.01	0.05±0.01
PA-CRF	41.33±0.05	64.27±0.01	31.66±0.10	58.21±0.01
KE-PN	**74.75±0.02**	**81.63±0.01**	**71.36±0.03**	**80.05±0.03**
Dataset: ACE2005				
Method	5-way 1-shot	5-way 5-shot	10-way 1-shot	10-way 5-shot
Match	23.50±0.46	35.56±0.03	15.28±0.74	33.43±0.12
Proto	22.45±0.05	50.01±0.07	18.47±0.03	45.05±0.03
Proto-dot	48.41±0.03	64.43±0.03	43.25±0.03	59.03±0.05
Relation	14.04±0.01	18.27±0.01	8.70±0.01	9.86±0.01
PA-CRF	48.20±0.02	65.13±0.02	40.70±0.08	60.68±0.01
KE-PN	**52.43±0.09**	**69.81±0.18**	**48.86±0.08**	**65.90±0.05**

Table 4: F1 scores (%) on all tasks and on three benchmark datasets: FewEvent, MAVEN and ACE2005. The best results among all models are marked in bold, which indicates statistically significant improvements over the best baseline with $p < 0.01$ under a boostrap test, and ± marks the standard deviation.

adopt a 3-layer MLP classifier, whose hidden size is 768. For event type relationship classification, we employ a 3-layer GCN, whose hidden size is also 768. Furthermore, for the CNN decoder, the hidden size is 768×2, the kernel size k is 3 and padding is 1. KE-PN is trained with the 1e-5 learning rate with the AdamW optimizer. We train KE-PN with 10,000 iterations on the training set and evaluate its performance with 3,000 iterations on the test set following the episodic paradigm (Vinyals et al., 2016), with batch size 1. Moreover, the dropout is 0.1. We run all experiments using PyTorch 1.5.1 on the Nvidia V100 GPU with 32GB memory.

5.3 Baseline Models

In the experiments, we adopt the representative and state-of-the-art joint models as baselines in order to verify the effectiveness of KE-PN on different tasks. More specifically, we choose the following baselines, which employ BERT as their base encoder:

1) Match (Vinyals et al., 2016), adopts Cosine similarity as the distance function;

2) Proto (Snell et al., 2017), uses Euclidean distance as the similarity metric;

3) Proto-dot, the Proto method that uses dot product to calculate the similarity;

4) Relation (Sung et al., 2018), adopts a two-layer neural network to measure the similarity;

5) PA-CRF (Cong et al., 2021), the state-of-the-art model on FewEvent by now.

Method	FewEvent		MAVEN	
	1-shot	5-shot	1-shot	5-shot
KE-PN	44.35	62.21	69.32	79.03
−ETRC	42.56	59.70	66.78	77.97
−SSL	39.32	56.14	61.79	72.20
−KE	28.13	44.39	37.94	59.19
−SSL	40.98	58.07	63.45	74.18
−KE	30.18	46.53	40.64	62.27
−ETRC	28.13	44.39	37.94	59.19

Table 5: The results of the ablation study on 5-way tasks on the dev sets of FewEvent and MAVEN.

5.4 Experimental Results

Table 4 presents the overall experimental results, where KE-PN outperforms all baselines and

Dataset	#Class	#Rel	#Avg Rel in 5-way	#Avg Rel in 10-way
FewEvent	100	334	0.67	3.03
ACE2005	33	103	1.95	8.77
MAVEN	100	61	0.12	0.55

Table 6: The statistic of relationships in three datasets.

achieves the state-of-the-art performance on all datasets and tasks. The F1 score of KE-PN increases by 16-30% on FewEvent, comparing to those baseline models. Furthermore, the improvements on MAVEN and ACE2005 are about 17-33% and 4-8%, respectively. The improvement on MAVEN is larger than the other two datasets, which is probably due to the strong association between MAVEN and FrameNet. The overall experimental results clearly demonstrate that KE-PN is effective on different datasets and tasks.

5.5 Ablation Study

In this subsection, we conduct ablation studies to investigate the effectiveness and impact of, both Knowledge-Enhanced Self-Supervised Learning (KESSL) and Event Type Relationship Classification (ETRC), as well as their impacts on the performance of KE-PN on the dev sets of FewEvent and MAVEN. Moreover, KESSL can be further divided into Knowledge Enhancement (KE) and Self-Supervised Learning (SSL). As shown in Table 5, the performance of ablated models without KE, SSL or ETRC consistently falls on all tasks. It suggests that all KE, SSL and ETRC contribute to the effectiveness of KE-PN. Besides, it can be observed that the improvement brought by ETRC is relatively small, which may be due to the sparsity of relationships in many tasks, as we count in Table 6. For example, the average number of relationships in 5-way tasks on MAVEN is only 0.12, which indicates that most 5-way tasks do not even include the concerned relationships. As a result, KESSL plays a more important role in KE-PN than ETRC.

5.6 In-depth Analysis

5.6.1 Visualization

To investigate the effectiveness and impact of ETRC, we adopt the Embedding projector[1] to visualize a 5-way 5-shot task on PA-CRF and KE-PN without KESSL, as shown in Figure 4. The sibling

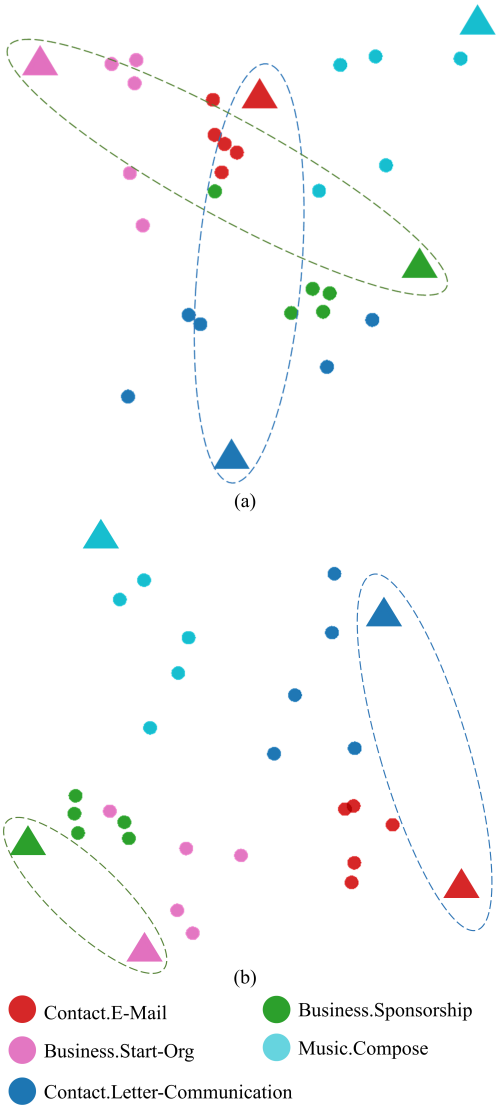

Figure 4: Visualization of instances and their prototypes of PA-CRF (a) and KE-PN without KESSL (b). The dots denote the instance embeddings of B-EventType and the triangles indicate the prototype representations of B-EventType.

prototypes (***Contact.E-mail*** and ***Contact.Letter-Communication***, ***Business.Start-Org*** and ***Business.Sponsorship***) produced by KE-PN without KESSL are closer to each other, compared with those by PA-CRF. In addition, embeddings produced by KE-PN without KESSL can be more easily separated at the level of meta types, compare to those by PA-CRF. For example, in the figure the ***Contact*** and ***Business*** meta types are linearly separable for KE-PN without KESSL, while linearly inseparable for PA-CRF. These observations indicate that ETRC can better capture the relevance between event types, and thus help obtain better performance.

[1] http://projector.tensorflow.org/

Event: ***Music.Compose***	
PA-CRF	In the development of European classical music, the function of composing (*B-Music.Compose*) music (*I-Music.Compose*) initially did not have much greater importance than that of performing it.
KE-PN w/o ETRC	In the development of European classical music, the function of composing (*B-Music.Compose*) music initially did not have much greater importance than that of performing it.
Event: ***Contact.E-Mail***	
PA-CRF	He says that 20 % of the people who get (*B- Contact.E-Mail*) that card send (*B- Contact.E-Mail*) him an e-mail.
KE-PN w/o ETRC	He says that 20 % of the people who get that card send (*B- Contact.E-Mail*) him an e-mail.

Table 7: The case study on PA-CRF and KE-PN without ETRC. The blue label denotes the right answers, and the red one indicates the wrong answers.

5.6.2 Case Study

To illustrate the effectiveness of KESSL, we choose two cases of event types ***Music.Compose*** and ***Contact.E-Mail*** from the FewEvent test set. As shown in Table 7, KE-PN without ETRC correctly predicts all labels of tokens on both two instances. Nevertheless, the baseline PA-CRF wrongly classifies the word "*music*" to ***I-Music.Compose*** and "*get*" to ***B-Contact.E-Mail***. It indicates that KESSL can help the model more effectively distinguish between trigger words and non-trigger words.

6 Conclusion and Future Work

In this paper, we proposed a novel knowledge-enhanced self-supervised prototypical network, called KE-PN, for FSED. KE-PN proposes hybrid rules which align the event types to FrameNet and then introduces knowledge to obtain more instances. Furthermore, KE-PN presents a novel self-supervised learning method to filter out noise from enhanced instances. Moreover, KE-PN adopts event type relationship classification as an auxiliary module, to inject relationship information into prototype representations. Extensive experiments on three benchmark FSED datasets, i.e., FewEvent, MAVEN and ACE2005, demonstrate the state-of-the-art performance of KE-PN. In the future work, we will explore FSED into a lifelong learning architecture, as the continuous FSED is an important problem in the real world.

7 Limitations

The limitations of KE-PN lie in two aspects, i.e., the method aspect and the resource aspect. From the method aspect, the hybrid rules by now are designed from an literal view, which can be explored to conduct semantic matching in the future work. In addition, we only take parent-child and sibling relationships into account in KE-PN, where more relationships between event types should be further studied. From the resource aspect, the GPU memory usage for training KE-PN increases due to the instances enhancement. So as to reduce the GPU memory usage, the batch size is set to 1, which may cause the learning process to be unstable.

Acknowledgments

This work is partially supported by the National Natural Science Foundation of China under grants U1911401, 62002341, and 61772501, the GFKJ Innovation Program, and the Lenovo-CAS Joint Lab Youth Scientist Project.

References

Collin F. Baker, Charles J. Fillmore, and John B. Lowe. 1998. The Berkeley FrameNet project. In *36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Volume 1*, pages 86–90, Montreal, Quebec, Canada. Association for Computational Linguistics.

Jiawei Chen, Hongyu Lin, Xianpei Han, and Le Sun. 2021. Honey or poison? solving the trigger curse in few-shot event detection via causal intervention. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8078–8088.

Yubo Chen, Liheng Xu, Kang Liu, Daojian Zeng, and Jun Zhao. 2015. Event extraction via dynamic multi-pooling convolutional neural networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 167–176.

Xin Cong, Shiyao Cui, Bowen Yu, Tingwen Liu, Wang Yubin, and Bin Wang. 2021. Few-shot event detection with prototypical amortized conditional random field. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 28–40.

Shiyao Cui, Bowen Yu, Tingwen Liu, Zhenyu Zhang, Xuebin Wang, and Jinqiao Shi. 2020. Edge-enhanced graph convolution networks for event detection with syntactic relation. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2329–2339.

Shumin Deng, Ningyu Zhang, Jiaojian Kang, Yichi Zhang, Wei Zhang, and Huajun Chen. 2020. Meta-learning with dynamic-memory-based prototypical network for few-shot event detection. In *Proceedings of the 13th International Conference on Web Search and Data Mining*, pages 151–159.

Shumin Deng, Ningyu Zhang, Luoqiu Li, Chen Hui, Tou Huaixiao, Mosha Chen, Fei Huang, and Huajun Chen. 2021. Ontoed: Low-resource event detection with ontology embedding. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2828–2839.

George R Doddington, Alexis Mitchell, Mark A Przybocki, Lance A Ramshaw, Stephanie M Strassel, and Ralph M Weischedel. 2004. The automatic content extraction (ace) program-tasks, data, and evaluation. In *Lrec*, volume 2, pages 837–840. Lisbon.

G David Forney. 1973. The viterbi algorithm. *Proceedings of the IEEE*, 61(3):268–278.

A Fritzler, V Logacheva, and M Kretov. 2019. Few-shot classification in named entity recognition task. In *Proceedings of the ACM Symposium on Applied Computing*, pages 993–1000.

Jonathan Gordon, John Bronskill, Matthias Bauer, Richard E Turner, Jan Stühmer, and Sebastian Nowozin. 2019. Meta-learning probabilistic inference for prediction. In *International Conference on Learning Representations*.

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

Viet Dac Lai, Minh Van Nguyen, Thien Huu Nguyen, and Franck Dernoncourt. 2021. Graph learning regularization and transfer learning for few-shot event detection. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2172–2176.

Viet Dac Lai, Thien Huu Nguyen, and Franck Dernoncourt. 2020. Extensively matching for few-shot learning event detection. In *Proceedings of the First Joint Workshop on Narrative Understanding, Storylines, and Events*, pages 38–45.

Shulin Liu, Yang Li, Feng Zhang, Tao Yang, and Xinpeng Zhou. 2019. Event detection without triggers. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 735–744.

Xiao Liu, Zhunchen Luo, and He-Yan Huang. 2018. Jointly multiple events extraction via attention-based graph information aggregation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1247–1256.

George A Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

Fionn Murtagh. 1991. Multilayer perceptrons for classification and regression. *Neurocomputing*, 2(5-6):183–197.

Thien Nguyen and Ralph Grishman. 2018. Graph convolutional networks with argument-aware pooling for event detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. 2009. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80.

Shirong Shen, Tongtong Wu, Guilin Qi, Yuan-Fang Li, Gholamreza Haffari, and Sheng Bi. 2021. Adaptive knowledge-enhanced bayesian meta-learning for few-shot event detection. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2417–2429.

Jake Snell, Kevin Swersky, and Richard Zemel. 2017. Prototypical networks for few-shot learning. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 4080–4090.

Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip HS Torr, and Timothy M Hospedales. 2018. Learning to compare: Relation network for few-shot learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1199–1208.

Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. 2016. Matching networks for one shot learning. *Advances in neural information processing systems*, 29.

Xiaozhi Wang, Ziqi Wang, Xu Han, Wangyi Jiang, Rong Han, Zhiyuan Liu, Juanzi Li, Peng Li, Yankai Lin, and Jie Zhou. 2020. Maven: A massive general domain event detection dataset. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1652–1671.