Few-Shot Event Detection with Prototypical Amortized Conditional Random Field

Xin Cong[1,2] **Shiyao Cui**[1,2] **Bowen Yu**[1,2]
Tingwen Liu[1,2*] **Yubin Wang**[1,2] and **Bin Wang**[3]
[1]Institute of Information Engineering, Chinese Academy of Sciences. Beijing, China
[2]School of Cyber Security, University of Chinese Academy of Sciences. Beijing, China
[3]Xiaomi AI Lab, Xiaomi Inc., Beijing, China
{congxin, cuishiyao, yubowen}@iie.ac.cn
{liutingwen, wangyubin}@iie.ac.cn
wangbin11@xiaomi.com

Abstract

Event detection tends to struggle when it needs to recognize novel event types with a few samples. The previous work attempts to solve this problem in the identify-then-classify manner but ignores the trigger discrepancy between event types, thus suffering from the error propagation. In this paper, we present a novel unified model which converts the task to a few-shot tagging problem with a double-part tagging scheme. To this end, we first propose the **P**rototypical **A**mortized **C**onditional **R**andom **F**ield (PA-CRF) to model the label dependency in the few-shot scenario, which approximates the transition scores between labels based on the label prototypes. Then Gaussian distribution is introduced for modeling of the transition scores to alleviate the uncertain estimation resulting from insufficient data. Experimental results show that the unified models work better than existing identify-then-classify models and our PA-CRF further achieves the best results on the benchmark dataset FewEvent. Our code and data are available at http://github.com/congxin95/PA-CRF.

1 Introduction

Event detection (ED) systems extract events of specific types from the given text. Traditionally, researchers use pipeline approaches (Ahn, 2006) where a trigger identification (TI) system is used to identify event triggers in a sentence and then a trigger classifier (TC) is used to find the event types of extracted triggers. Such a framework makes the task easy to conduct but ignores the interaction and correlation between the two subtasks, being susceptible to cascading errors. In the last few years, several neural network-based models were proposed to jointly identify triggers and classify event types from a sentence (Chen et al., 2015; Nguyen and Grishman, 2015, 2018; Liu et al., 2018; Yan et al., 2019; Cui et al., 2020b,a). These models have achieved promising performance and proved the effectiveness of solving ED in the joint framework. But they almost followed the supervised learning paradigm and depended on the large-scale human-annotated dataset, while new event types emerge every day and most of them suffer from the lack of sufficient annotated data. In the case of insufficient resources, existing joint models cannot recognize the novel event types with only few samples, i.e., Few-Shot Event Detection (FSED).

Marry	It served as the location of Bogart's [wedding] to Bacall.
E-Mail	If you have a better idea, please [e-mail] me.

Figure 1: An example from FewEvent dataset revealing the trigger discrepancy. "[·]" marks the event trigger.

One intuitive way to solve this problem is to first identify event triggers in the conventional way and then classify the event types based on the few-shot learning (Vinyals et al., 2016; Snell et al., 2017; Sung et al., 2018), these two subtasks can be trained jointly by parameter sharing. Such identify-then-classify paradigm (Deng et al., 2020) seems to be convincing because TI aims to recognize triggers and does not need to adapt to novel classes, so we just need to solve the TC in the few-shot manner. Unfortunately, our preliminary experiments reveal that TI tends to struggle when recognizing triggers of novel event types because novel events usually contain completely different triggers with the semantic distinction from the known events, i.e., **Trigger discrepancy** problem. Figure 1 gives an example that the trigger "e-mail" would only occur in event *E-Mail* but not in *Marry* and triggers of two events have disparate context. And experiments on FewEvent (a benchmark dataset for FSED) show that 59.21% triggers in the test set do not trigger

[*]Corresponding Author

any events in the training set and the F1 score of TI with the SOTA TI model BERT-tagger (Yang et al., 2019) is only 31.06%. Thus, the performance of the identify-then-classify paradigm will be limited by the TI part due to the cascading errors.

In this paper, we present a new unified method to solve FSED. Specifically, we convert this task to a sequence labeling problem and design a double-part tagging scheme using trigger and event parts to describe the features of each word in a sentence. The key to the sequence labeling framework is to model the dependency between labels. Conditional Random Field (CRF) is a popular choice to capture such label dependency by learning transition scores of fixed label space in the training dataset. Nevertheless, in FSED, CRF cannot be applied directly due to the **label discrepancy** problem, that is the label space of the test set is non-overlapping with the training set since FSED aims to recognize novel event types. Therefore, the learned transition scores of CRF from the training set do not model the dependency of the novel labels in the test set.

To address the label discrepancy problem, we propose **P**rototypical **A**mortized **C**onditional **R**andom **F**ield (PA-CRF), which approximates the transition scores based on the label prototypes (Snell et al., 2017) instead of learning by optimization. Specifically, we first apply the self-attention mechanism to capture the dependency information between labels and then map the label prototype pairs to the corresponding transition scores. In this way, PA-CRF can produce label-specific transition scores based on the few supportive samples, which can adapt to arbitrary novel event types. However, predicting the transition score as a single fixed value actually acts as the point estimation, which usually acquires a large amount of annotated data to achieve accurate estimation. Estimated from the handful of samples, the transition scores may suffer from the statistical uncertainty due to the random fluctuation of scant data. To release this issue, inspired by variational inference (Kingma and Welling, 2014; Yoon et al., 2018; Gordon et al., 2019), we treat the transition score as the random variable and utilize the Gaussian distribution to approximate its distribution to model the uncertainty. Thus, our PA-CRF is to estimate the parameters of the Gaussian distribution rather than the transition scores directly, i.e., in the amortized manner (Kingma and Welling, 2014; Gordon et al., 2019). The Probabilistic Inference (Gordon et al., 2019) is employed based on the Gaussian distribution to make the inference robust by taking the possible perturbation of transition scores into account since the perturbation is also learned in a way that coherently explains the uncertainty of the samples.

To summarize, our contributions are as follows:

- We devise a tagging-based unified model for FSED. To the best of our knowledge, we are the first to solve this task in a unified manner, free from the cascading errors.
- We propose a novel model, PA-CRF, which estimates the distributions of transition scores for modeling the specific label dependency in the few-shot sequence labeling setting.
- Experimental results show that our proposed PA-CRF outperforms other competitive baselines on the FewEvent dataset. Further analyses show the effectiveness of our unified model and the limitation of the identify-then-classify models.

2 Related Work

Few-shot Event Detection Event Detection (ED) aims to recognize the specific type of events in a sentence. In recent years, various neural-based models have been proposed and achieved promising performance in ED (Chen et al., 2015; Nguyen and Grishman, 2015, 2018; Liu et al., 2018; Yan et al., 2019; Cui et al., 2020b). Chen et al. (2015) and Nguyen and Grishman (2015) proposed the convolution architecture to capture the semantic information in the sentence. Nguyen et al. (2016) introduced the recurrent neural network to model the sequence contextual information of words. Recently, GCN-based models (Nguyen and Grishman, 2018; Liu et al., 2018; Yan et al., 2019; Cui et al., 2020b) have been proposed to exploit the syntactic dependency information and achieved state-of-the-art performance. However, all these models are data-hungry, limiting dramatically their usability and deployability in real-world scenarios.

Recently, there has been an increasing research interest in solving event detection in the few-shot scenarios (Deng et al., 2020; Lai et al., 2020a,b), by exploiting the Few-Shot Learning (Vinyals et al., 2016; Snell et al., 2017; Finn et al., 2017; Sung et al., 2018; Cong et al., 2020). Lai et al. (2020a) proposed LoLoss which splits the part of the support set to act as the auxiliary query set to train the model. Lai et al. (2020b) introduced two regularization matching losses to improve the performance of models. These works only focus on the few-shot trigger classification which classifies the event type of the annotated trigger according to the context based on few samples. This is unrealistic as triggers of novel events are predicted by some existing toolkits in advance. Deng et al. (2020) first proposed the benchmark dataset, *FewEvent*, for FSED and designed the DMBPN based on the dynamic memory networks. They train a conventional trigger identifier and a few-shot trigger classifier jointly and evaluated the model performance in the identify-then-classify paradigm. Moreover, our preliminary experiments reveal that the conventional trigger identification model tends to struggle when recognizing triggers of novel event types because of the trigger discrepancy between different event types. Thus, errors of the trigger identifier might be propagated to the event classification. Different from the previous identify-then-classify framework, for the first time, we solve Few-Shot Event Detection with two subtasks in a unified manner.

Few-shot Sequence Labeling In recent years, several works (Fritzler et al., 2019; Hou et al., 2020; Yang and Katiyar, 2020) have been proposed to solve the few-shot named entity recognition using sequence labeling methods. Fritzler et al. (2019) applied the vanilla CRF in the few-shot scenario directly. Hou et al. (2020) proposed a collapsed dependency transfer mechanism (CDT) into CRF, which learns label dependency patterns of a set of task-agnostic abstract labels and utilizes these patterns as transition scores for novel labels. Yang and Katiyar (2020) trains their model on the training data in a standard supervised learning manner and then uses the prototypical networks and the CDT for prediction in the inference phase. Different from these methods learning the transition scores by optimization, we build a network to generate the transition scores based on the label prototypes instead. In this way, we can generate exact label-specific transition scores of arbitrary novel event types to achieve adaptation ability. And we further introduce the Gaussian distribution to estimate the data uncertainty. Experiments prove the effectiveness of our method over the previous methods.

3 Problem Formulation

We convert event detection to a sequence labeling task. Each word is assigned a label that contributes to detecting the events. Labels consist of two parts: the word position in the trigger and the event type. We use the "BI" (Begin, Inside) signs to represent the position information of a word in the event trigger. The event type information is obtained from a predefined set of events. Label "O" (Other) means that the corresponding word is independent of the target events. Thus, the total number of labels is $2N + 1$ (N for *B-EventType*, N for *I-EventType*, and an additional *O* label), where N is the number of predefined event types.

Furthermore, we formulate the Few-Shot Event Detection in the typical N-way-K-shot paradigm. Let $\boldsymbol{x} = \{w_1, w_2, \ldots, w_n\}$ denote an n-word sequence, and $\boldsymbol{y} = \{y_1, y_2, \ldots, y_n\}$ denote the label sequence of the $\boldsymbol{x}$. Given a *support set* $\mathcal{S} = \{(\boldsymbol{x}^{(i)}, \boldsymbol{y}^{(i)})\}_{i=1}^{N \times K}$ which contains N event types and each event type has only K instances, FSED aims to predict the labels of a unlabeled *query set* $\mathcal{Q}$ based on the *support set* $\mathcal{S}$. Formally, a $\{\mathcal{S}, \mathcal{Q}\}$ pair is called a N-way-K-shot task $\mathcal{T}$. There exist two datasets consisting of a set of tasks : $\mathcal{D}_{train} = \{\mathcal{T}^{(i)}\}_{i=1}^{M_{train}}$ and $\mathcal{D}_{test} = \{\mathcal{T}^{(i)}\}_{i=1}^{M_{test}}$ where M_{train} and M_{test} denote the number of the task in two datasets respectively. As the name suggests, $\mathcal{D}_{train}$ is used to train models in the training phase while $\mathcal{D}_{test}$ is for evaluation. It is noted that these two datasets have their own event types, which means that the label space of two datasets is disjoint with each other.

4 Methodology

4.1 Overview

As described above, we formulate FSED as the few-shot sequence labeling task with interdependent labels. Following the widely used CRF framework, we propose a novel PA-CRF model to model such label dependency in the few-shot setting, and decode the best-predicted label sequence. Our PA-CRF contains three modules: 1) Emission Module: It first computes the prototype of each label based on the support set, and then calculates the similarity between prototypes and each token in the query set as the emission scores. 2) Transition Module: It exploits the prototypes to generate the parameters of Gaussian distribution of the transition scores for decoding. 3) Decoding Module: Based on the emission scores and Gaussian distributed transition scores, the Decoding Module calculates the probabilities of possible label sequences for the given query set and decodes the predicted label sequence.

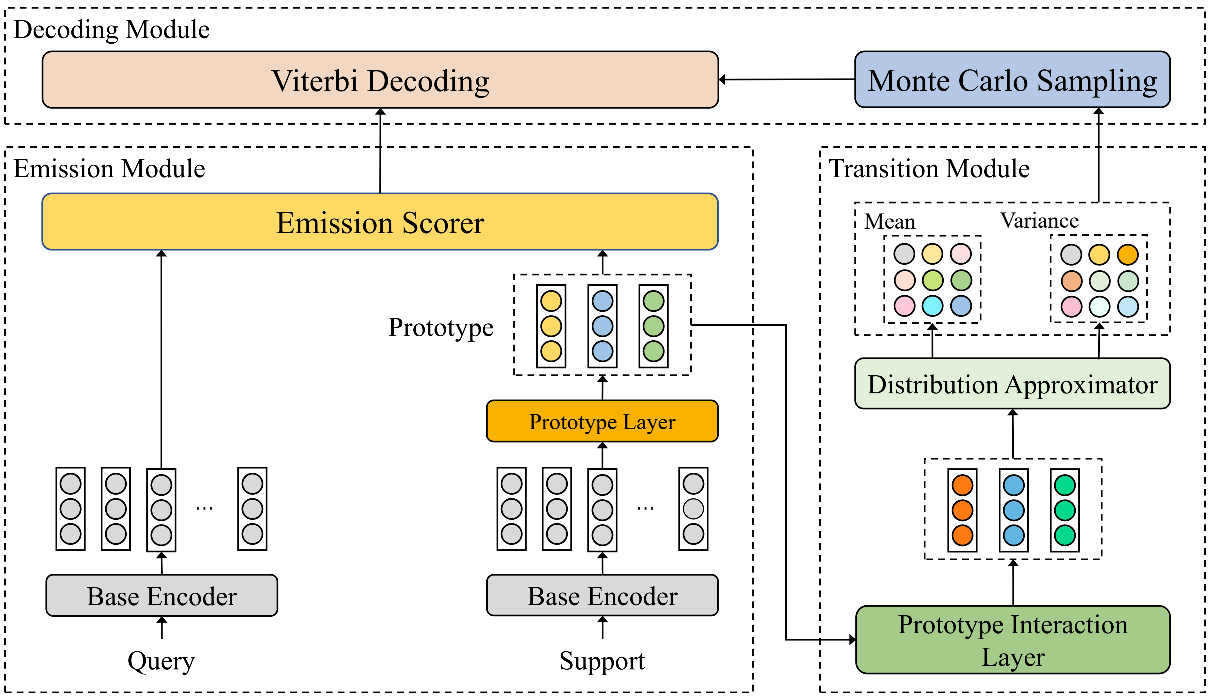

Figure 2: Architecture of our proposed PA-CRF. It consists of three modules: a) Emission Module calculates the emission scores for the query instance based on the prototypes derived from the support set. b) Transition Module generates the Gaussian distributed transition scores with respect to prototypes. c) Decoding Module exploits the emission scores and approximated Gaussian distributed transition scores to decode the predicted label sequence with the Monte Carlo Sampling.

Figure 2 gives an illustration of PA-CRF. We detail each component from the bottom to the top.

4.2 Emission Module

The Emission Module assigns the emission scores to each token of sentences in the query set $\mathcal{Q}$ with regard to each label based on the support set $\mathcal{S}$.

4.2.1 Base Encoder

Base Encoder aims to embed tokens in both support set $\mathcal{S}$ and query set $\mathcal{Q}$ into real-value embedding vectors to capture the semantic information.

Since BERT (Devlin et al., 2019) shows its advanced ability to capture the sequence information and has been widely used in NLP tasks recently, we use it as the backbone. Given an input word sequence $\boldsymbol{x}$, BERT first maps all tokens into hidden embedding representations. We denote this operation as:

$$\{\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_n\} = \mathrm{BERT}(\boldsymbol{x}) \tag{1}$$

where $\mathbf{h}_i \in \mathbb{R}^{d_h}$ refers to the hidden representation of token w_i, d_h is the dimension of the hidden representation.

4.2.2 Prototype Layer

Prototype Layer is to derive the prototypes of each label from the support set $\mathcal{S}$. As described in the problem formulation, we use the BIO schema to annotate the event trigger and N event types could contain $2N+1$ labels. Thus, indeed, we could get $2N+1$ prototypes. Following the previous work (Snell et al., 2017), we calculate the prototype of each label by averaging all the word representations with that label in the support set $\mathcal{S}$:

$$\mathbf{c}_i = \frac{1}{|\mathcal{S}(y_i)|} \sum_{w \in \mathcal{S}(y_i)} \mathbf{h}, \quad i = 1, 2, \ldots, 2N+1, \tag{2}$$

where $\mathbf{c}_i$ denotes the prototype for label y_i, $\mathcal{S}(y_i)$ refers to the token set containing all words in the support set $\mathcal{S}$ with label y_i, $\mathbf{h}$ represents the corresponding hidden representation of token w, and $|\cdot|$ is the number of set elements.

4.2.3 Emission Scorer

Emission Scorer aims to calculate the emission score for each token in the query set $\mathcal{Q}$. The emission scores are calculated according to the similarities between tokens and prototypes. The computation of the emission score of the label y_i for the word w_j is defined as:

$$f_E(y_i, w_j, \mathcal{S}) = d(\mathbf{c}_i, \mathbf{h}_j), \tag{3}$$

where $d(\cdot, \cdot)$ is the similarity function. In practice, we choose the dot product operation to measure the similarity.

Finally, given a word sequence $\boldsymbol{x}$, the emission score of the whole sentence with its corresponding ground-truth label sequence $\boldsymbol{y}$ is computed as:

$$\text{EMIT}(\boldsymbol{y}, \boldsymbol{x}, \mathcal{S}) = \sum_{i=1}^{n} f_E(y_i, w_i, \mathcal{S}). \tag{4}$$

4.3 Transition Module

In vanilla CRF, the transition scores are learnable parameters and optimized from large-scale data to model the label dependency. However, in the few-shot scenarios, the learned transition scores cannot adapt to the novel label set due to the disjoint label space. To overcome this problem, we use neural networks to generate the transition scores based on the label prototypes instead of learning transition scores by optimization to achieve adaptation ability. In this case, a problem needing to be solved is that using few support instances with random data fluctuation to generate transition scores would cause uncertain estimation and result in wrong inference. To model the uncertainty, we treat the transition score as a random variable and use the Gaussian distribution to approximate its distribution. Specifically, the Transition Module is to generate the distributional parameters (mean and variance) of transition scores based on the label prototypes. It consists of two layers: 1) Prototypical Interaction Layer and 2) Distribution Approximator. Details of each layer are listed in the following parts.

4.3.1 Prototype Interaction Layer

Since the transition score is to model the dependency between labels, the individual prototype for each event type with rare dependency information is hard to generate their transition scores. Thus, we propose a Prototype Interaction Layer which exploits the self-attention mechanism to capture the dependency between labels.

We first calculate the attention scores of each prototype $\mathbf{c}_i$ with others:

$$\alpha_{ij} = \frac{\exp(\mathbf{c}_i^{(q)} \cdot \mathbf{c}_j^{(k)})}{\sum_{m=1}^{2N+1} \exp(\mathbf{c}_i^{(q)} \cdot \mathbf{c}_m^{(k)})}, \tag{5}$$

where $\mathbf{c}_i^{(q)}$ and $\mathbf{c}_i^{(k)}$ are transformed from $\mathbf{c}_i$ by two linear layers respectively:

$$\begin{aligned} \mathbf{c}_i^{(q)} &= W^{(q)}\mathbf{c}_i + b^{(q)} \\ \mathbf{c}_i^{(k)} &= W^{(k)}\mathbf{c}_i + b^{(k)} \end{aligned} \tag{6}$$

Getting the attention scores, the prototype $\tilde{\mathbf{c}}_i$ with dependency information is calculated as follows:

$$\tilde{\mathbf{c}}_i = \sum_{j=1}^{2N+1} \alpha_{ij}\mathbf{c}_j^{(v)}, \tag{7}$$

where $\mathbf{c}_i^{(v)}$ is also transformed linearly from $\mathbf{c}_i$:

$$\mathbf{c}_i^{(v)} = W^{(v)}\mathbf{c}_i + b^{(v)} \tag{8}$$

4.3.2 Distribution Approximator

This module aims to generate the mean and variance of Gaussian distributions based on the prototypes with dependency information.

Given the label set $\mathcal{Y}$ with total $2N + 1$ labels, we first denote the transition score matrix as $T_r \in \mathbb{R}^{(2N+1)\times(2N+1)}$ for all label pairs, and denote the the i-th row j-th column element of T_r as $[T_r]_{ij}$ which refers to the transition score for i-th label transiting to j-th label in the label set $\mathcal{Y}$. As treating $[T_r]_{ij}$ as random variable, we use the Gaussian distribution $[\tilde{T_r}]_{ij} \sim \mathcal{N}(\mu_{ij}, \sigma_{ij}^2)$ to approximate $[T_r]_{ij}$, where $\mathcal{N}(\cdot, \cdot)$ refers to the Gaussian distribution. To estimate the mean μ_{ij} and variance σ_{ij} of $[\tilde{T_r}]_{ij}$, we concatenate the corresponding prototypes $\tilde{\mathbf{c}}_i$ and $\tilde{\mathbf{c}}_j$ and feed into two feed-forward neural networks respectively:

$$\mu_{ij} = W^{(\mu)}\left[\tilde{\mathbf{c}}_i \| \tilde{\mathbf{c}}_j\right] + b^{(\mu)} \tag{9}$$

$$\sigma_{ij}^2 = \exp\left(W^{(\sigma^2)}\left[\tilde{\mathbf{c}}_i \| \tilde{\mathbf{c}}_j\right] + b^{(\sigma^2)}\right) \tag{10}$$

where $[\cdot\|\cdot]$ means the concatenation operation. Given a label sequence $\boldsymbol{y}$, the transition score of the whole label sequence is approximated by:

$$\text{TRANS}(\boldsymbol{y}, \tilde{T_r}) = \sum_{i=1}^{n-1} [\tilde{T_r}]_{\mathbb{I}(y_i)\mathbb{I}(y_{i+1})} \tag{11}$$

where $\mathbb{I}(y_i)$ refers to the label index in $\mathcal{Y}$ of y_i.

4.4 Decoding Module

Decoding Module derives the probabilities for a specific label sequence of the query set according to the emission scores and approximated Gaussian distributions of transition scores.

Since the approximated transition score is Gaussian distributional and not a single value, we denote the probability density function of the approximated transition score matrix as $q(\tilde{T_r}|\mathcal{S})$. According to the Probabilistic Inference (Gordon et al., 2019), the probability of label sequence $\boldsymbol{y}$ of a word sequence $\boldsymbol{x}$ based on the support set $\mathcal{S}$ is calculated as:

$$P(\boldsymbol{y}|\boldsymbol{x},\mathcal{S}) = \int P(\boldsymbol{y}|\boldsymbol{x},\mathcal{S},\tilde{T}_r)q(\tilde{T}_r|\mathcal{S})\mathrm{d}\tilde{T}_r \tag{12}$$

Following the CRF algorithm, the probability can be calculated based on the Equation 4 and Equation 11:

$$P(\boldsymbol{y}|\boldsymbol{x},\mathcal{S}) = \int \frac{1}{Z}\exp\left(\mathrm{EMIT}(\boldsymbol{y},\boldsymbol{x},\mathcal{S}) + \mathrm{TRANS}(\boldsymbol{y},\tilde{T}_r)\right) q(\tilde{T}_r|\mathcal{S})\mathrm{d}\tilde{T}_r \tag{13}$$

where

$$Z = \sum_{\boldsymbol{y'}\in Y} \exp\left(\mathrm{EMIT}(\boldsymbol{y'},\boldsymbol{x},\mathcal{S}) + \mathrm{TRANS}(\boldsymbol{y'},\tilde{T}_r)\right) \tag{14}$$

and Y refers to all possible label sequences.

In the training phase, we use negative log-likelihood loss as our objective function:

$$\mathcal{L} = -\mathop{\mathbb{E}}_{(\boldsymbol{x},\boldsymbol{y})\sim\mathcal{Q}}[\log P(\boldsymbol{y}|\boldsymbol{x},\mathcal{S})] \tag{15}$$

Due to the hardness to compute the integral of Equation 13, in practice, we use the Monte Carlo sampling technique (Gordon et al., 2019) to approximate the integral. To make the sampling process differentiable for optimization, we employ the reparameterization trick (Kingma and Welling, 2014) for each transition score $[\tilde{T}_r]_{ij}$:

$$[\tilde{T}_r]_{ij} = \mu_{ij} + \epsilon\sigma_{ij}, \text{where } \epsilon \sim \mathcal{N}(0,1) \tag{16}$$

In the inference phase, the Viterbi algorithm (Forney, 1973) is employed to decode the best-predicted label sequence for the query set.

5 Experiment

5.1 Dataset

We conduct experiments on the benchmark *FewEvent* dataset introduced in the previous work (Deng et al., 2020), which is the currently largest few-shot dataset for event detection. It contains 70,852 instances for 100 event types and each event type owns about 700 instances on average. Since Deng et al. (2020) do not share their split train/dev/test set, we re-split the FewEvent in the same ratio as Deng et al. (2020). We use 80 event types as the training set, 10 event types as the dev set, and the rest 10 event types as the test set. More statistics of FewEvent dataset are listed in Appendix A.

5.2 Evaluation

We follow the evaluation metrics in previous event detection works (Chen et al., 2015; Liu et al., 2018; Cui et al., 2020b), an event trigger is marked correct if and only if its event type and its offsets in the sentence are both correct. We adopt the standard micro F1 score to evaluate the results and report the averages and standard deviations over 5 randomly initialized runs.

6 Implementation Details

We employ *BERT-BASE-UNCASED* (Devlin et al., 2019) as the base encoder. The maximum sentence length is set as 128. Our model is trained using *AdamW* optimizer with the learning rate of 1e-5. All the hyper-parameters are tuned on the dev set manually. In the training phase, we follow the widely used episodic training (Vinyals et al., 2016) in few-shot learning. Episodic training aims to mimic N-way-K-shot scenario in the training phase. In each epoch, we randomly sample N event types from the training set and each event type randomly sample K instances as support set and other M instances as the query set. We train our model with 20,000 iterations on the training set and evaluate its performance with 3,000 iterations on the test set following the episodic paradigm. We run all experiments using PyTorch 1.5.1 on the Nvidia Tesla T4 GPU, Intel(R) Xeon(R) Silver 4110 CPU with 256GB memory on Red Hat 4.8.3 OS.

6.1 Baselines

To investigate the effectiveness of our proposed method, we compare it with a range of baselines and state-of-the-art models, which can be categorized into three classes: fine-tuning paradigm, identify-then-classify paradigm and unified paradigm.

Fine-tuning paradigm solves the FSED in the standard supervised learning, i.e., pre-training on the large scale dataset and fine-tuning on the handful target data. We adopt the state-of-the-art model, **PLMEE** (Yang et al., 2019), of the standard ED into the FSED directly.

Identify-then-classify models first perform trigger identification (named as TI) and then classify the event types based on the few-shot learning methods (named as FSTC). We investigate two typed of identify-then-classify paradigms: separate and multi-task. For the separate models, the trigger identifier and few-shot trigger classifier are

Paradigm	Model	5-Way-5-Shot	5-Way-10-Shot	10-Way-5-Shot	10-Way-10-Shot
Fine-tuning	PLMEE	4.43 ± 0.19	4.69 ± 0.85	2.52 ± 0.28	2.76 ± 0.55
Separate	LoLoss	30.14 ± 0.30	30.91 ± 0.29	29.33 ± 0.40	30.08 ± 0.39
	MatchLoss	29.78 ± 0.14	30.75 ± 0.15	28.75 ± 0.23	29.59 ± 0.21
Multi-task	LoLoss	31.51 ± 1.56	31.70 ± 1.21	30.46 ± 1.38	30.32 ± 0.89
	MatchLoss	30.44 ± 0.99	30.68 ± 0.78	28.97 ± 0.61	30.05 ± 0.93
	DMBPN	37.51 ± 2.60	38.14 ± 2.32	34.21 ± 1.45	35.31 ± 1.69
Unified	Match	39.93 ± 1.67	46.02 ± 1.20	30.88 ± 1.08	35.91 ± 1.19
	Proto	50.11 ± 0.77	52.97 ± 0.95	43.51 ± 1.16	42.70 ± 0.98
	Proto-Dot	58.82 ± 0.88	61.01 ± 0.23	55.04 ± 1.62	58.78 ± 0.88
	Relation	28.91 ± 1.13	29.83 ± 0.78	18.49 ± 1.25	21.47 ± 1.40
	Vanilla CRF	59.01 ± 0.81	62.21 ± 1.94	56.00 ± 1.51	59.35 ± 1.09
	CDT	<u>59.30</u> ± 0.23	<u>62.77</u> ± 0.12	<u>56.41</u> ± 1.09	<u>59.44</u> ± 1.83
	StructShot	57.69 ± 0.91	61.54 ± 1.23	54.54 ± 0.95	57.14 ± 0.79
	PA-CRF	**62.25*** ± 1.42	**64.45*** ± 0.49	**58.48*** ± 0.68	**61.64*** ± 0.81

Table 1: F1 scores (10^{-2}) of different models on the FewEvent test set. Bold marks the highest number among all models, underline marks the second-highest number, and ± marks the standard deviation. * marks statistically significant improvements over the best baseline with $p < 0.01$ under a boostrap test.

trained separately without parameter sharing. We first exploit the state-of-the-art BERT-tagger for the TI task. It uses BERT (Devlin et al., 2019) and a linear layer to tag the trigger in the sentence as a sequence labeling task. Since TI just aims to recognize the occurrence of the trigger, the label set only contains three labels: *O*, *B-Trigger*, *I-Trigger*. For the FSTC task, we reimplement two FSTC models: **LoLoss** (Lai et al., 2020a), **MatchLoss** (Lai et al., 2020b). In the multi-task models, we reimplement **DMBPN** (Deng et al., 2020) and replace its encoder with BERT for the fair comparison. DMBPN combines a conventional trigger identification module and a few-shot trigger classification module by parameter sharing. But in the inference phase, it detects the event trigger still in the identify-then-classify paradigm. Additionally, we also provide the multi-task version of the LoLoss and MatchLoss which are trained jointly with BERT-tagger with shared BERT parameters.

Unified models perform few-shot event detection with a single model without task decomposition. Because we are the first to solve this task in a unified way, there is no previous unified model that can be compared. But for the comprehensive evaluation of our proposed PA-CRF model, we also construct two groups of variants of PA-CRF: non-CRF models and CRF-based models. Non-CRF models use emission scores to predict via softmax and do not take the label dependency into consideration. We implement four typical few-shot classifiers: 1) **Match** (Vinyals et al., 2016) uses cosine function to measure the similarity, 2) **Proto** (Snell et al., 2017) uses Euclidean Distance as the similarity metric, 3) **Proto-Dot** uses dot product to compute the similarity, 4) **Relation** (Sung et al., 2018) builds a two-layer neural networks to measure the similarity. Since CRF with the capacity of modeling label dependency is widely used in sequence labeling task, we implement three kinds of CRF-based models as our baselines: 1) **Vanilla CRF** (Fritzler et al., 2019): We adopt the vanilla CRF in the FSED task without considering the adaptation problem. 2) **CDT** (Hou et al., 2020): As the SOTA of the few-shot NER task, we reimplement it according to the official code and adapt it in the FSED task to replace our Transition Module. 3) **StructShot** (Yang and Katiyar, 2020): It is also a few-shot NER model. It first pre-trains on the training set and utilizes the prototypical networks and the CDT for prediction based on the support set in the inference phase. For the fair comparison, the emission module of these CRF-based baseline models is the same as our PA-CRF.

6.2 Main Results

Table 1 summarizes the results of our PA-CRF against other baseline models on the FewEvent test set.

Comparison with fine-tuning model It is obvious that PLMEE performs poorly in all four few-shot settings and all few-shot-based methods outperform it with an absolute gap, which powerfully proves that the conventional supervised methods is incapable of solving FSED.

Comparison with identify-then-classify models (1) Most of unified models (except Relation) perform higher than all identify-then-classify models, especially for PA-CRF with huge gaps about 30%, proving the effectiveness of the unified framework. (2) Comparing with the separate paradigm, the multi-task paradigm is able to improve performance but it still cannot catch up with the unified paradigm. (3) DMBPN works better than other two models but still works poorly to handle the FSED due to the limitation of the TI. We will discuss the bottleneck of the identify-then-classify paradigm in Section 6.3.

Comparison with unified models (1) Over the best non-CRF baseline model Proto-Dot, PA-CRF achieves substantial improvements of 3.43%, 3.44%, 3.44% and 2.86% on four few-shot scenarios respectively, which confirms the effectiveness and rationality of PA-CRF to model the label dependency. (2) Vanilla CRF performs better than other non-CRF baseline methods, which demonstrates that CRF is able to improve the performance by modeling the label dependency, even if the learned transition scores do not match the label space of the test set. (3) Compared to Vanilla CRF, both CDT and StructShot achieve slightly higher F1 scores, indicating the transition scores of abstract BIO labels can improve the model adaptation ability to some extent. (4) CDT exceeds the StructShot since CDT is trained based on the episodic training, which makes it learns the class-agnostic token representations. (5) PA-CRF outperforms CDT (2.95%, 1.68%, 2.07% and 2.20% in four few-shot settings respectively) with absolute gaps. We consider that it is because CDT learning the transition scores of the abstract labels cannot model the exact dependency of specific label set, so its adaptation ability is limited. In contrast, PA-CRF generates the label-specific transition scores based on the label prototype, which can capture the dependency for specific novel event types. (6) Comparing four few-shot scenarios, we can find that the F1 score increases as the K-shot increases, which shows that more support samples can provide more information of the event type. The F1 score decreases as the N-way increases when the shot number is fixed, which reveals that the larger way number causes more event types to predict which increases the difficulty of the correct detection.

Model	TI	FSTC	FSED
LoLoss	31.06	95.27	30.14
DMBPN	40.64	95.44	37.51
DMBPN(CDT-TI)	54.69	95.49	53.93
PA-CRF	63.68	96.76	62.25

Table 2: Comparison of PA-CRF and baselines on two subtasks. F1 scores are reported on the FewEvent test set in the 5-way-5-shot setting.

To summarize, we can draw the conclusion that (1) The identify-then-classify paradigm is incapable of solving the FSED task. (2) Compared to the identify-then-classify paradigm, the unified paradigm works more effectively for the FSED task. (3) Approximating transition scores based on the label prototypes not by optimization, our PA-CRF achieves better adaptation on novel event types.

6.3 Bottleneck Analysis

To investigate the bottleneck of the identify-then-classify paradigm, we evaluate LoLoss (separate model), DMBPN (multi-task model) and PA-CRF (unified model) on two subtasks: TI and FSTC separately in the 5-way-5-shot setting on the FewEvent test set. To reduce the influence of the cascading errors, we use the ground truth trigger span for evaluation in the FSTC. The experimental results are reported in Table 2. From Table 2, we find that: (1) All models achieve more than 95% F1 score on the FSTC task, indicating that both identify-then-classify and unified models is capable enough of solving the FSTC problem. (2) For the TI task, two identify-then-classify baselines perform 31.06% and 40.64% F1 score respectively, which demonstrates that the conventional TI module has difficulty in adapting to novel event triggers. Hence, due to the cascading errors, the poorly-performed TI module limits the performance of the identify-then-classify models. (3) PA-CRF achieves 63.68% F1 score on TI task, which exceeds the two kinds of identify-then-classify models significantly. Unlike identify-then-classify models recognizing triggers based on seen triggers, PA-CRF utilizes the trigger representations from the support set of the novel event types to identify novel triggers so our unified

Model	5-Shot	10-Shot
PA-CRF	44.39	51.06
- Distribution Estimation	43.47	49.41
- Interaction Layer	41.62	45.74
- Transition Score	39.83	45.07

Table 3: Ablation study of PA-CRF in 5-Way settings. F1 scores are reported on the FewEvent dev set.

model works better in the TI task of FSED. In conclusion, the conventional trigger identifier cannot identify novel triggers in FSED, and exploiting the support set of novel event types is necessary.

6.4 Effectiveness Analysis

To verify the effectiveness of the unified framework, we adapt our best baseline model, CDT, to replace TI module of DMBPN to solve trigger identification in the few-shot manner. It identifies triggers based on the emission scores between tokens and label prototypes calculating from the support set and learned abstract transition scores. In this case, we rename it as DMBPN(CDT-TI) and evaluate it in TI and FSTC subtasks. Results are also reported based on the 5-way-5-shot setting in Table 2 and we observe that: The CDT-TI-based DMBPN achieve 54.69% in TI task, exceeding the conventional TI based models, which shows that solving TI in the few-shot manner by utilizing the support set can reduce the trigger discrepancy to some extent. Although the performance of FSTC is similar to the original DMBPN, owing to the improvements of TI task, the final performance of FSED exceeds the original DMBPN by 16.42% but they are still inferior to PA-CRF with a huge gap (8.99% on TI task). Therefore, we draw the conclusion that solving FSED in the unified manner can utilize the correlation between two subtasks to improve the model performance significantly.

6.5 Ablation Study

To study the contribution of each component in our PA-CRF model, we run the ablation study on the FewEvent dev set. From these ablations (see Table 3), we find that: (1) - Distribution Estimation: To study whether distributional estimation is helpful to improve the performance, we remove it and make the Distribution Approximator generate a single value as the transition score directly as the point estimation. And the inference is based on the generated transition scores without Probabilistic Inference. As a result, the F1 score drops 1.02% and 1.65% in two scenarios, respectively. We attribute these gaps to our proposed Gaussian-based distributional estimation which can model the data fluctuation to relieve the influence of data uncertainty. (2) - Interaction Layer: To certify that the Prototype Interaction Layer contributes to capturing the information between prototypes, we remove it and evaluate in two scenarios. We read from Table 3 that F1 scores decrease significantly by 2.77% and 5.32% respectively, which indicates that the Prototype Interaction Layer is able to capture the dependency among prototypes. (3) - Transition Score: To prove the contribution of the label dependency, we remove the Transition Module and only use the emission score for prediction. Results show that without transition scores, the performance of the model drops dramatically by 4.56% and 5.99% respectively, which powerfully proves that the transition score can improve the performance of the few-shot sequence labeling task.

Furthermore, we have conducted case study and error analysis to validate the strength of our PA-CRF and explore its weakness. Details are listed in Appendix B and Appendix C.

7 Conclusion

In this paper, we explore a new viewpoint of solving few-shot event detection in a unified manner. Specifically, we propose a prototypical amortized conditional random field to generate the transition scores to achieve adaptation ability for novel event types based on the label prototypes. Furthermore, we present the Gaussian-based distributional estimation to approximate transition scores to relieve the statistical uncertainty of data fluctuation. Finally, experimental results on the benchmark FewEvent dataset prove the effectiveness of our proposed method. In the future, we plan to adapt our method to other few-shot sequence labeling tasks such as named entity recognition.

Acknowledgements

We would like to thank all reviewers for their insightful comments and suggestions. This work is supported in part by the Strategic Priority Research Program of Chinese Academy of Sciences (grant No. XDC02040400) and the Youth Innovation Promotion Association of CAS (Grant No. 2021153).

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, Sydney, Australia. Association for Computational Linguistics.

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, Beijing, China. Association for Computational Linguistics.

Xin Cong, Bowen Yu, Tingwen Liu, Shiyao Cui, Hengzhu Tang, and Bin Wang. 2020. Inductive unsupervised domain adaptation for few-shot classification via clustering.

Shiyao Cui, Bowen Yu, Xin Cong, Tingwen Liu, Quangang Li, and Jinqiao Shi. 2020a. Label enhanced event detection with heterogeneous graph attention networks. *CoRR*, abs/2012.01878.

Shiyao Cui, Bowen Yu, Tingwen Liu, Zhenyu Zhang, Xuebin Wang, and Jinqiao Shi. 2020b. Edge-enhanced graph convolution networks for event detection with syntactic relation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings, EMNLP*.

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 *WSDM '20: The Thirteenth ACM International Conference on Web Search and Data Mining*.

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

Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning, ICML*.

G. D. Forney. 1973. The viterbi algorithm. *Proceedings of the IEEE*.

Alexander Fritzler, Varvara Logacheva, and Maksim Kretov. 2019. Few-shot classification in named entity recognition task.

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

Yutai Hou, Wanxiang Che, Yongkui Lai, Zhihan Zhou, Yijia Liu, Han Liu, and Ting Liu. 2020. Few-shot slot tagging with collapsed dependency transfer and label-enhanced task-adaptive projection network. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1381–1393, Online. Association for Computational Linguistics.

Diederik P. Kingma and Max Welling. 2014. Auto-encoding variational bayes. In *2nd International Conference on Learning Representations, ICLR*.

Viet Dac Lai, Franck Dernoncourt, and Thien Huu Nguyen. 2020a. Exploiting the matching information in the support set for few shot event classification. In *Advances in Knowledge Discovery and Data Mining - 24th Pacific-Asia Conference, PAKDD*.

Viet Dac Lai, Franck Dernoncourt, and Thien Huu Nguyen. 2020b. Extensively matching for few-shot learning event detection. *CoRR*, abs/2006.10093.

Xiao Liu, Zhunchen Luo, and Heyan 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, Brussels, Belgium. Association for Computational Linguistics.

Thien Huu Nguyen, Kyunghyun Cho, and Ralph Grishman. 2016. Joint event extraction via recurrent neural networks. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Thien Huu Nguyen and Ralph Grishman. 2015. Event detection and domain adaptation with 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 2: Short Papers)*, pages 365–371, Beijing, China. Association for Computational Linguistics.

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

Jake Snell, Kevin Swersky, and Richard Zemel. 2017. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems*.

Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip H. S. Torr, and Timothy M. Hospedales. 2018. Learning to compare: Relation network for few-shot learning. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR*.

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

Haoran Yan, Xiaolong Jin, Xiangbin Meng, Jiafeng Guo, and Xueqi Cheng. 2019. Event detection with multi-order graph convolution and aggregated attention. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5766–5770, Hong Kong, China. Association for Computational Linguistics.

Fan Yang, Xiaochang Peng, Gargi Ghosh, Reshef Shilon, Hao Ma, Eider Moore, and Goran Predovic. 2019. Exploring deep multimodal fusion of text and photo for hate speech classification. In *Proceedings of the Third Workshop on Abusive Language Online*, pages 11–18, Florence, Italy. Association for Computational Linguistics.

Yi Yang and Arzoo Katiyar. 2020. Simple and effective few-shot named entity recognition with structured nearest neighbor learning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP*.

Jaesik Yoon, Taesup Kim, Ousmane Dia, Sungwoong Kim, Yoshua Bengio, and Sungjin Ahn. 2018. Bayesian model-agnostic meta-learning. In *Advances in Neural Information Processing Systems*.

A Dataset Statistics

	Training Set	Dev Set	Test Set
# Cls.	80	10	10
# Sent.	67982	2173	697
# Tok./sent	36.5	38.6	30.8

Table 4: Statistics of FewEvent Dataset.

Table 4 lists the statistics of FewEvent dataset containing the number of event type (#Cls.), the number of sentence(# Sent.), the number of token per sentence (# Tok./sent) for the train/dev/test set.

Figure 3 demonstrates the data imbalance problem of FewEvent dataset. Event "Marry" has the most instance (26135 instances) while event "E-Mail" only has 30 instances. 69% event types have less than 100 instances while 7% event types have more than 1000 instances. However, since we use episodic training (Vinyals et al., 2016) to train our model, the data imbalance problem can be relieved to some extent.

B Case Study

We compare our method with the best identify-then-classify baseline, DMBPN and the best unified baseline, CDT in some cases, as shown in Table 5.

As demonstrated by the first example of a *Sponsorship* event, DMBPN, in the identify-then-classify paradigm, fails to identify the trigger *sponsorship*. According to our statistics about the FewEvent dataset, 95.16% triggers of *Sponsorship* event do not occur in the training set. Since DMBPN uses the conventional TI module which is trained on the training set to identify the event trigger, it is incapable of identifying the *Sponsorship* event trigger. Although the classification module of DMBPN succeeds to distinguish the event type as *Sponsorship*, due to the cascading errors, the final prediction of event trigger (containing the span and type) is incorrect. As a result, the performance of DMBPN as an identify-then-classify model on the FSED task is limited. In contrast, our unified PA-CRF is successful to detect the event trigger *sponsorship* of this case since PA-CRF utilizes the information of the support set of *Sponsorship* event in which word *sponsorship* appears and acts the trigger.

In the second example, the best unified baseline, CDT, tags the first trigger word *locked* with *I-Jail* label wrongly. That is because CDT learns the abstract transition scores among a set of abstract labels which cannot model the label dependency for this specific event type accurately. Thanks to the PA-CRF which models the label dependency based on the label prototypes from the support set of *Jail* event, our model is capable of tagging the word *locked* with *B-Jail* label correctly.

C Error Study

Although our method outperforms all baseline models, we still observe some failure cases. Table 6 gives a typical example of the wrong prediction of event *Transport* (*Trans* for short). For the query instance, the ground truth event trigger is "pouring out". The word "pouring" should be labeled as *B-Trans* and the *out* should be labeled as *I-Trans*. However, our model only detects "pouring" with *B-Trans* while missing "out". From the support set, we find that all support instances of this event type only contain the one-word trigger without *I-Trans* label tokens, resulting in that the prototype of *I-Trans* is zero vector. As a result, the emission score for the label *I-Trans* of each query token is calculated as zero and the transition scores based on the prototypes are also affected. Therefore, our model is not able to detect the *I-Trans* label correctly in this case. In the future, we will further study to

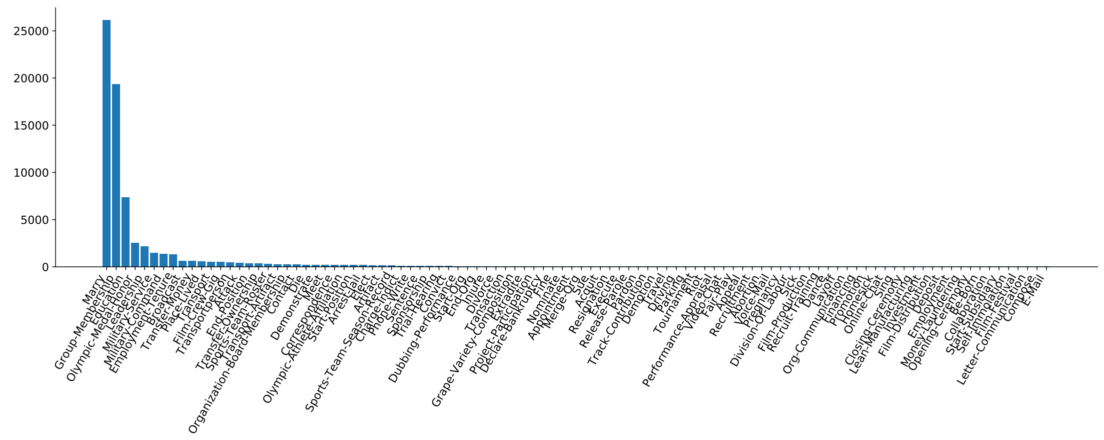

Figure 3: The data imbalance of FewEvent dataset.

Model	Prediction
DBMPN	Candlestick Park was dropped when the $[\textit{sponsorship}]_O$ agreement expired.
PA-CRF	Candlestick Park was dropped when the $[\textit{sponsorship}]_{B-Sponsorship}$ agreement expired.
CDT	Willmore will tell everyone for wanting to keep the poor man $[\textit{locked}]_{I-Jail}$ $[\textit{up}]_{I-Jail}$.
PA-CRF	Willmore will tell everyone for wanting to keep the poor man $[\textit{locked}]_{B-Jail}$ $[\textit{up}]_{I-Jail}$.

Table 5: Output of PA-CRF, DMBPN and CDT on samples from the FewEvent test set. The subscripts denote the labels tagged by the models.

Support #1	Cult members $[\textit{visited}]_{B-Trans}$ and built a laser weapon mounted on a truck
Support #2	Israel $[\textit{leave}]_{B-Trans}$ the West Bank and Gaza and dismantle Jewish settlements.
Truth	Refugees have been $[\textit{pouring}]_{B-Trans}$ $[\textit{out}]_{I-Trans}$ of Fallujah over the last few days.
Prediction	Refugees have been $[\textit{pouring}]_{B-Trans}$ $[\textit{out}]_O$ of Fallujah over the last few days.

Table 6: A case of the wrong prediction from the FewEvent test set. The subscripts denote the triggers and their event types. We only list two support instances to reduce space.

solve the missing *I* label problem.

D Analyses about Various Dataset Split

Model	R1	R2	R3	R4	R5
PA-CRF	59.0	33.4	53.1	42.4	48.0
DMBPN	44.9	31.1	40.8	32.6	27.9

Table 7: Performance of our PA-CRF and DMBPN in various split FewEvent dataset in the 5-Way-5-Shot scenario. F1 scores (10^{-2}) are reported.

Since Deng et al. (2020) do not public their split train/dev/test set of FewEvent dataset, to compare our PA-CRF with DMBPN (Deng et al., 2020), we re-split the FewEvent randomly in the same split ratio as the Deng et al. (2020) (80 event types for training set, 10 event types for dev set and the rest 10 event types for test set) and evaluate the DBMPN performance on our split test set. However, in our experiments, the performance of DMBPN is lower than the original paper. We assume that the different data split may influence the model performance badly. To validate our assumption, we re-split the FewEvent dataset for five random seeds and conduct more experiments on these various split train/dev/test set. The results are reported in Table 7. From Table 7, it can be observed that: (1) Data split does influence the model performance significantly indeed. In these five different split train/dev/test set, the performance of PA-CRF varies from 59.0% to 33.4% with a huge range. Similarly, DMBPN also varies from 44.9% to 27.9%, owning a huge gap about 20%. It demonstrates that for the FewEvent dataset, different split could cause huge fluctuation of the model performance. Therefore, our PA-CRF including baselines performs lower than those Deng et al. (2020) reported due to the different data split. (2) PA-CRF outperforms DMBPN in all five random split settings, which powerfully proves the robustness of PA-CRF over the identify-then-classify paradigm.