Prototype-based Prompt-Instance Interaction with Causal Intervention for Few-shot Event Detection

Jingyao Tang[1], **Lishuang Li**[1]*, **Hongbin Lu**[2], **Xueyang Qin**[1]
Beibei Zhang[1], **Haiming Wu**[3]
[1] Dalian University of Technology, Dalian, China
[2] Liaoning Normal University, Dalian, China
[3] Beijing Institute of Technology, Beijing, China
tangjingyao@mail.dlut.edu.cn, lils@dlut.edu.cn, qinxueyang@snnu.edu.cn, hm.wu@bit.edu.cn
{lulu163163, luhongbin-123}@163.com

Abstract

Few-shot Event Detection (FSED) is a meaningful task due to the limited labeled data and expensive manual labeling. Some prompt-based methods are used in FSED. However, these methods require large GPU memory due to the increased length of input tokens caused by concatenating prompts, as well as additional human effort for designing verbalizers. Moreover, they ignore instance and prompt biases arising from the confounding effects between prompts and texts. In this paper, we propose a prototype-based prompt-instance **Inter**action with causal **Inter**vention (**2xInter**) model to conveniently utilize both prompts and verbalizers and effectively eliminate all biases. Specifically, 2xInter first presents a Prototype-based Prompt-Instance Interaction (PPII) module that applies an interactive approach for texts and prompts to reduce memory and regards class prototypes as verbalizers to avoid design costs. Next, 2xInter constructs a Structural Causal Model (SCM) to explain instance and prompt biases and designs a Double-View Causal Intervention (DVCI) module to eliminate these biases. Due to limited supervised information, DVCI devises a generation-based prompt adjustment for instance intervention and a Siamese network-based instance contrasting for prompt intervention. Finally, the experimental results show that 2xInter achieves state-of-the-art performance on RAMS and ACE datasets.

Keywords: Few-shot event detection, Prompt learning, Causal intervention

1. Introduction

Event Detection (ED) aims to extract specific event types from text with provided trigger words. We usually consider ED as supervised learning, which requires a substantial amount of annotated data (Walker et al., 2005). However, the labeled data is insufficient and manual annotation is costly. As a result, researchers have shifted their focus toward Few-Shot Event Detection (FSED).

Recently, some researchers conduct FSED using prototype-based methods (Lai et al., 2020a, 2021; Deng et al., 2020; Zhang et al., 2022; Zhao et al., 2022), which aim to learn a metric space that measures the distance between instances and prototypes for prediction. Although they are effective for few-shot data, they can be sensitive to inter-task data distribution variations.

Another powerful way for FSED is prompt-based methods (Song et al., 2023a,b; Li et al., 2022; Yue et al., 2023), which transform downstream tasks into ones that pre-trained language models are familiar with, thereby relaxing distribution constraints about prototypical approaches. However, these methods usually concatenate original text and prompt, increasing input length and exacerbating GPU memory usage. Meanwhile, devising appropriate verbalizers can be challenging (see Fig. 1(a)).

Figure 1: (a) An example of the use of prompts and verbalizers. (b) An example of confounding effects.

Additionally, in prompt-based models, the confounding effects between instances and prompts may result in biases (see Fig. 1(b)). On the one hand, instances may mislead the effect of prompts on predictions, which we call *instance bias*. For example, when learning the effect of the prompt "*talked* and *Entity* of *[MASK]*" on the correct result "*Contact:Phone-Write*", the feature of the instance "*I've talked to people in Pennsylvania.*" may lead to the wrong result "*Contact:Meet*". On the other hand, prompts may disturb the impact of instances

* Corresponding author

LREC-COLING 2024, pages 13269–13278

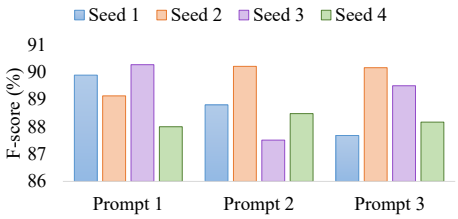

Figure 2: The performance of 4 seeds using 3 different prompts on the ACE dataset for a 5+1-way 5-shot setting.

on predictions, which we refer to as *prompt bias*. For example, an altered prompt "*talked . Entity . [MASK]*" might lead a model to predict the incorrect category "*Contact:Online-Chat*". Moreover, we visualize the prompt bias in Fig. 2, which shows that different prompts with the same instance yield different outcomes. Therefore, these biases can incur poor stability and robustness for FSED.

In this paper, we propose a prototype-based prompt-instance **Inter**action with causal **Inter**vention (**2xInter**) model to mitigate the above problems. Firstly, we present a Prototype-based Prompt-Instance Interaction (PPII) module to efficiently and conveniently employ prompts and verbalizers. Specifically, the module uses an interactive method rather than a concatenating approach to combine prompts and instances, which reduces the GPU memory usage. Meanwhile, class prototypes serve as verbalizers in the module, which avoids design difficulties.

Secondly, from a causal view, we establish a prompt-based Structural Causal Model (SCM) to explain the instance bias and prompt bias. Subsequently, we propose a Double-View Causal Intervention (DVCI) module to eliminate these biases. Moreover, traditional methods usually employ backdoor adjustments for causal intervention, but such adjustments are difficult to implement because of the limited supervised information. As a result, we devise a generation-based prompt adjustment for instance intervention and a Siamese network-based instance contrasting for prompt intervention.

We conduct experiments on three benchmark FSED datasets: RAMS (Ebner et al., 2020), ACE (Walker et al., 2005), and LR-KBP (Lai et al., 2021). Experimental results show that 2xInter consistently attains State-of-the-Art (SOTA) results across most datasets and few-shot settings, thus verifying the validity of our model. The main contributions of this paper are as follows:

- We propose a novel FSED model called 2xInter. Experimental results demonstrate that 2xInter[1] achieves SOTA performance on RAMS and ACE datasets.
- We present a Prototype-based Prompt-Instance Interaction (PPII) module, which leverages prompt information in a memory-saving manner and utilizes prototypical verbalizer in a labor-saving design.
- We construct a Structural Causal Model (SCM) to explain the causes of instance bias and prompt bias. Then we propose a Double-View Causal Intervention (DVCI) module to de-bias them. In DVCI, we devise a generation-based prompt adjustment for instance intervention and a Siamese network-based instance contrasting for prompt intervention.

[1] https://github.com/manderous/2xInter

2. Methodology

In this section, we introduce the prototype-based prompt-instance **inter**action with causal **inter**vention (**2xInter**) model, whose overall framework is presented in Fig. 3. 2xInter consists of three modules: Prototype-based Prompt-Instance Interaction (PPII), Structural Causal Model (SCM), and Double-View Causal Intervention (DVCI).

2.1. Task definition

In this paper, Few-Shot Event Detection (FSED) is framed as an "$N+1$-way K-shot" few-shot classification (Lai et al., 2020a). Here, $N+1$ represents the number of event types plus one "NULL" type, and K represents the number of samples for each category.

Moreover, X denotes event instances, R denotes actual labels, T denotes prompts, and Y denotes predicted labels. Since the input data is categorized into support set and query set, these variables can be represented as a concatenation of two variables, i.e., $X = [X^s, X^q]$, $R = [R^s, R^q]$, $T = [T^s, T^q]$, and $Y = [Y^s, Y^q]$. The superscript s annotates the support variables and q annotates the query variables. Additionally, $X^s = \left[x^s_1, \ldots, x^s_{(N+1)\cdot K}\right]$ and other support variables follow the same mathematical form. Similarly, $X^q = \left[x^q_1, \ldots, x^q_{(N+1)\cdot L}\right]$ and other query variables follow the same pattern. Here, L denotes the number of samples for each query category. Therefore, FSED is designed to predict the labels of the query samples based on the support samples.

2.2. Prototype-based Prompt-Instance Interaction

This section proposes the PPII module to efficiently integrate prompts and to conveniently use verbalizers. First, we utilize the interactive method to combine prompts and instances, considering that the concatenating method consumes more GPU

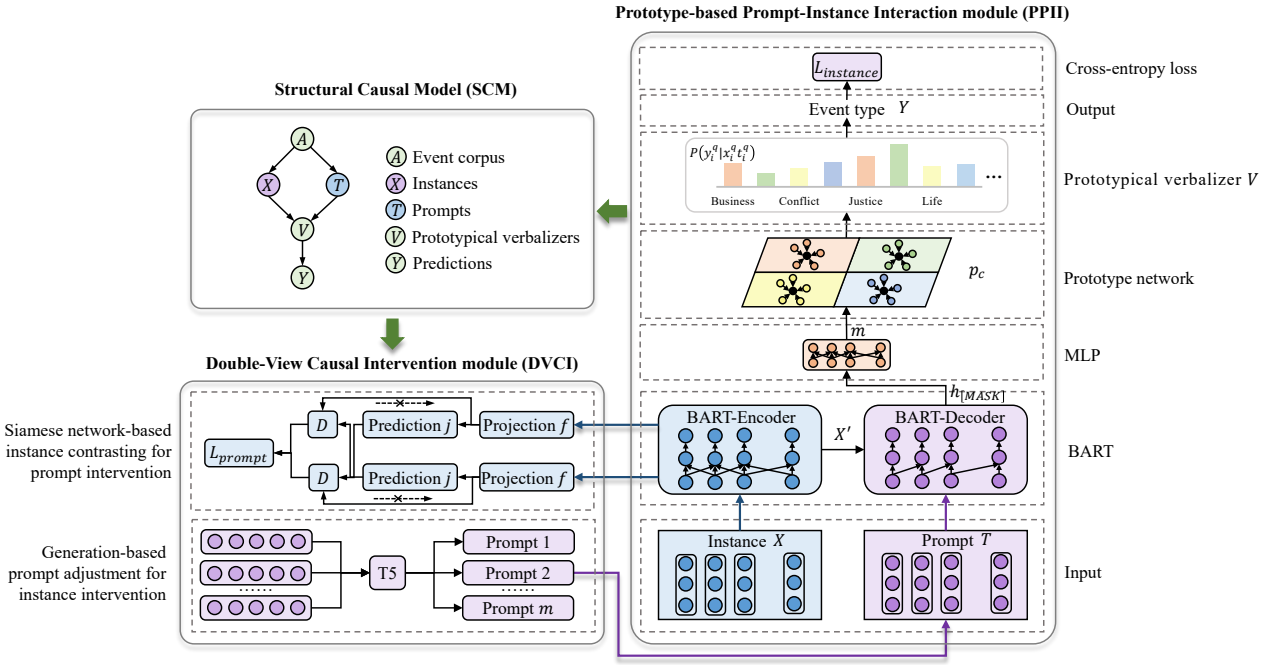

Figure 3: The overall framework of 2xInter. The PPII module aims to classify event types. Secondly, the SCM is built using the PPII module with the goal of analyzing biases. Finally, the MVCI module aims to eliminate the biases identified by the SCM.

memory. Specifically, the instances X is fed into the BART encoder (Lewis et al., 2020) f_{enc} and the output of the encoder interacts with the prompts T at the BART decoder f_{dec}. The equations corresponding to this process are shown below:

$$X' = f_{enc}(X), \; H = f_{dec}(T; X'), \tag{1}$$

where H means the instance-interacted prompt representations. Following these formulas, we extract the vector $h_{[MASK]}$ viewed as the final event representation from *[MASK]* position of H.

Second, we regard category prototypes as verbalizers to avoid design difficulties. Specifically, the event representation $h_{[MASK]}$ is initially fed into an MLP with parameter W:

$$M = W h_{[MASK]}. \tag{2}$$

Subsequently, we obtain the prototype representation p_c of each class c by averaging the representation $m_i^{s(c)} \in M$ of all support samples under that class: $p_c = \frac{1}{K} \sum_{i=i}^{K} m_i^{s(c)}, c = 0, 1, \ldots, N+1$. Then, we compute the predicted probability of each category y_i^q for a sample x_i^q and prompt t_i^q:

$$P(y_i^q | x_i^q, t_i^q) = \frac{exp(-d(m_i^q, p_{y_i}))}{\sum_{c=1}^{N+1} exp(-d(m_i^q, p_c))}, \tag{3}$$

where $d(\cdot)$ denotes a metric function for measuring the similarity between the query sample $m_i^q \in M$ and the prototype p_c. Eq. (3) serves as a prototypical verbalizer to obtain scores for each category.

Finally, we optimize the PPII module using cross-entropy loss:

$$L = -\sum_{i=1}^{(N+1)\cdot L} \sum_{c=1}^{N+1} [I(r_i^q = c) \cdot log P(y_i^q = c | x_i^q, t_i^q)], \tag{4}$$

where $r_i^q \in R^q$, $y_i^q \in Y^q$, $t_i^q \in T^q$, and $I(r_i^q = c)$ is an indicator function representing whether r_i^q is equal to c.

2.3. Structural causal model and bias analysis

In this section, we first establish a SCM based on the PPII module, and then use the SCM to analyze instance bias and prompt bias.

2.3.1. Structural causal model

We construct a PPII-based SCM, as shown in Fig. 4(a). This SCM comprises five variables: event corpus (A), instances (X), event prompts (T), prototypical verbalizers (V), and predictions (Y). Furthermore, the SCM includes three causal paths based on these five variables:

$X \leftarrow A \rightarrow T$: signifies that event instances and prompts are derived from event corpus.

$X \rightarrow V \leftarrow T$: indicates that the verbalizers are calculated from event instances and prompts.

$V \rightarrow Y$: represents that the results are obtained from the verbalizer.

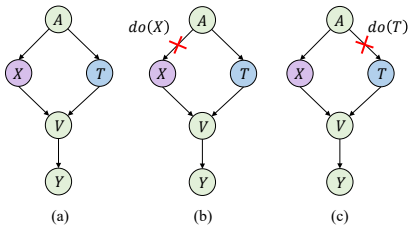

Figure 4: The structural causal model and causal intervention. (a) represents the structural causal model for PPII. (b) indicates the instance intervention. (c) denotes the prompt intervention. In addition, the red cross represents causal intervention.

2.3.2. Instance bias and prompt bias

We utilize the SCM to analyze all biases in the prompt-based models. First, X can confound the effect from T to Y due to the backdoor path $A \rightarrow X \rightarrow V$, which we refer to as instance bias. For instance, when learning the effect of the prompt "*talked* and *Entity* of *[MASK]*" on the correct result "*Contact:Phone-Write*", the characteristics of the instance "*I've talked to people in Pennsylvania, yes.*" might lead to the incorrect outcome "*Contact:Meet*" because of the backdoor path.

In addition, T can confound the effect from X to Y due to the backdoor path $A \rightarrow T \rightarrow V$, leading to what we term as prompt bias. In prompt learning, we assign a specific prompt to each instance. However, different instances are suitable for different prompts. For example, the instance "*I've talked to people from Pennsylvania, yes.*" favors prompt "*[Trigger]* and *[Argument type]* of *[MASK]*", while the instance "*we read all of your e-mail*" prefers prompt "*[Trigger] . [Argument type] . [MASK]*". To illustrate this issue, we select three semantically equivalent but differently expressed prompts, and train our PPII with these prompts under four different random seeds. The results are presented in Fig. 2, which demonstrates that different prompts yield varying performance rankings across these random seeds.

In summary, PPII contains the instance bias and prompt bias, which may incur poor stability and robustness for FSED.

2.4. Double-View causal interventions

In this section, we propose the DVCI module to eliminate the instance bias and prompt bias. Specifically, we intervene in the samples and prompts to block the two backdoor paths ($A \rightarrow X \rightarrow V$ and $A \rightarrow T \rightarrow V$) in the SCM, respectively, as shown in Fig. 4(b) and Fig. 4(c). Consequently, the loss function transforms from Eq. (4) into the following form:

$$
\begin{aligned}
L &= L_{instance} + L_{prompt} = \\
&- \sum_{i=1}^{(N+1)\cdot L} \sum_{c=1}^{N+1} \left[I\left(r_i^q = c\right) \cdot logP\left(y_i^q = c | do\left(x_i^q = x_0\right)\right) \right] \\
&- \sum_{i=1}^{(N+1)\cdot L} \sum_{c=1}^{N+1} \left[I\left(r_i^q = c\right) \cdot logP\left(y_i^q = c | do\left(t_i^q = t_0\right)\right) \right],
\end{aligned} \tag{5}
$$

where $do\left(\cdot\right)$ denotes causal intervention, while x_0 and t_0 denote the text and prompt of the current sample, respectively. Furthermore, we apply the backdoor adjustment to calculate the do operation:

$$
P\left(y_i^q | do\left(x_i^q = x_0\right)\right) = \textstyle\sum_{t_i^d \in T^d} P\left(t_i^q\right) P\left(y_i^q \middle| x_0, t_i^d\right), \tag{6}
$$

$$
P\left(y_i^q | do\left(t_i^q = t_0\right)\right) = \textstyle\sum_{x_i^d \in X^d} P\left(x_i^q\right) P\left(y_i^q | x_i^q, t_0\right), \tag{7}
$$

where T^d stands for all possible prompts for the current text, and X^d signifies all possible texts for the current prompt. However, it is challenging to traverse all possible instances and prompts. Therefore, we address this issue in the following section.

2.4.1. Generation-based prompt adjustment for instance intervention

It is intractable to iterate over all prompts for instance intervention (i.e., Eq. (6)). Therefore, we utilize the Text-to-Text Transfer Transformer (T5) (Raffel et al., 2020) to generate a sequence of prompts $T_d = \{t_1, t_2, \ldots, t_m\}$. Specifically, we initially format all samples as "*[Sentence] [Trigger] [U] [Argument type] [V]*" to input into the T5 model because we aim to acquire prompts containing both trigger words and argument types. In this input format, "*[Sentence]*" indicates the sentence, "*[Trigger]*" denotes the trigger words, and "*[Argument type]*" represents the argument type. In addition, "*[U]*" and "*[V]*" serve as placeholders, which will later be filled with specific words when they are outputted from T5. Following this, we view the top m texts generated by T5 as m prompts. We briefly visualize the generation process in Fig. 5.

Assuming that the tokens of each prompt are represented as $t_i = \left\{k_i^1, k_i^2, \ldots, k_i^{|t_i|}\right\}$, where $|t_i|$ represents the length of the tokens, the probability of generating the tokens by the T5 model is $l_i = \sum_{j=1}^{|t_i|} logP_{T5}\left(k_i^j \middle| k_i^1, \ldots, k_i^{j-1}\right)$, where P_{T5} represents the output probability distribution of the T5. Thus, the $P\left(t_i^q\right) c$ in Eq. (6) can be computed as follows:

$$
P\left(t_i^q\right) = \frac{exp\left(l_i\right)}{\sum_j exp\left(l_j\right)}. \tag{8}
$$

Additionally, $P\left(y_i^q \middle| x_0, t_i^d\right)$ in Eq. (6) can be calculated by Eq. (3). Therefore, we can obtain the loss $L_{instance}$ by generation-based prompt adjustment.

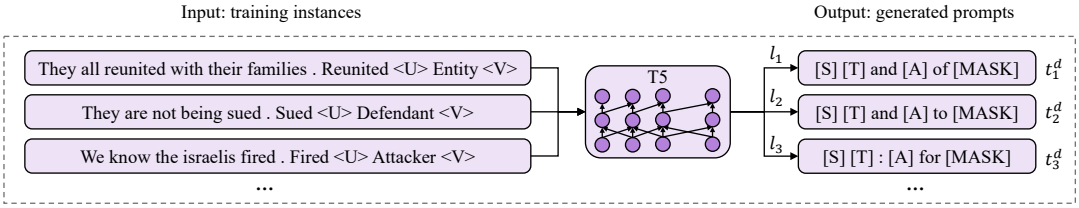

Figure 5: Prompt generation process, where [S], [T], and [A] serve as placeholders of a sentence, trigger word, and argument type respectively.

2.4.2. Siamese network-based instance contrasting for prompt intervention

We need to iterate through all instances for prompt intervention (i.e., Eq. (7)), but the supervised information is limited for few-shot tasks. Thus, we apply a surrogate method, which is a Siamese network-based instance contrasting, to compute Eq. (7). This method provides two positive instances for each original instance, thereby adding extra supervised information. Concretely, an event instance is fed into the BART-Encoder twice, resulting in two positive samples, x_i^1 and x_i^2, with different dropout masks in the two calculations:

$$x_i^1 = f_\theta\left(x_i, \phi_1\right), x_i^2 = f_\theta\left(x_i, \phi_2\right), \tag{9}$$

where $x_i \in X$, and ϕ_1 and ϕ_2 represent the different random masks used for dropout in BART, respectively. After that, the two positive instances are separately input into a projection MLP head g, which shares weights for the two instances:

$$z_1 = g\left(x_i^1\right), z_2 = g\left(x_i^2\right). \tag{10}$$

Furthermore, the outputs of g are passed through a prediction MLP head j, which converts a representation of a positive instance into that of another one:

$$p_1 = j\left(z_1\right), p_2 = j\left(z_2\right). \tag{11}$$

Finally, we optimize the model using the following surrogate loss function:

$$\begin{aligned} L_{prompt} = & \frac{1}{2} D\left(p_1, stopgrad\left(z_2\right)\right) \\ & + \frac{1}{2} D\left(p_2, stopgrad\left(z_1\right)\right), \end{aligned} \tag{12}$$

where $D\left(\cdot\right)$ stands for cross entropy, i.e., $D\left(p_1, z_2\right) = -softmax\left(z_2\right) \cdot logsoftmax\left(p_1\right)$. Therefore, we can get the loss L_{prompt} by Siamese network-based instance contrasting. Moreover, Fig. 6 illustrates the process of the Siamese network-based instance contrasting.

3. Experiments

3.1. Dataset

We evaluate our model on three datasets: RAMS (Ebner et al., 2020), ACE (Walker et al., 2005), and LR-KBP (Deng et al., 2020). LR-KBP integrates ACE-2005[2] and TAC-KBP-2017[3] datasets and expands some event types from Freebase (Bollacker et al., 2008) and Wikipedia (Milne and Witten, 2008) datasets. The partition details of all datasets are exhibited in Table 1.

Split	RAMS		ACE		LR-KBP	
	#C	#S	#C	#S	#C	#S
Train	95	5340	18	2865	72	6732
Dev	17	1934	11	1227	10	561
Test	22	1793	11	1226	10	1291

Table 1: Statistics of three datasets. #C and #S denote the number of classes and the number of samples, respectively.

3.2. Baselines

We divided the baselines utilized in our experiments into three groups and summarized them as follows:

Prototype-based methods comprise Proto (Snell et al., 2017), InterIntra (Lai et al., 2020a), DMB-Proto (Deng et al., 2020), ProAcT (Lai et al., 2021), HCL-TAT (Zhang et al., 2022), and KE-PN (Zhao et al., 2022). These methods tackle few-shot tasks using a prototype network. To maintain experimental fairness, we additionally fine-tune both Proto (the typical prototype-based model) and ProAct (the best-performing model) for comparison as our model is also fine-tuned. Furthermore, the results were reported for Proto, InterIntra, DMB-Proto, and ProAcT on both the BERTMLP (Yang et al., 2019) and BERTGCN (Lai et al., 2020b) sentence encoders.

Prompt-based methods include P4E (Li et al., 2022) and MetaEvent (Yue et al., 2023). These methods resolve few-shot event detection through prompt learning. Since MetaEvent reported results only on the LR-KBP dataset in the 10+1-way 10-shot setting, we employed its code to reproduce results for other datasets and settings.

[2] http://projects.ldc.upenn.edu/ace/
[3] https://tac.nist.gov/2017/KBP/Event/index.html

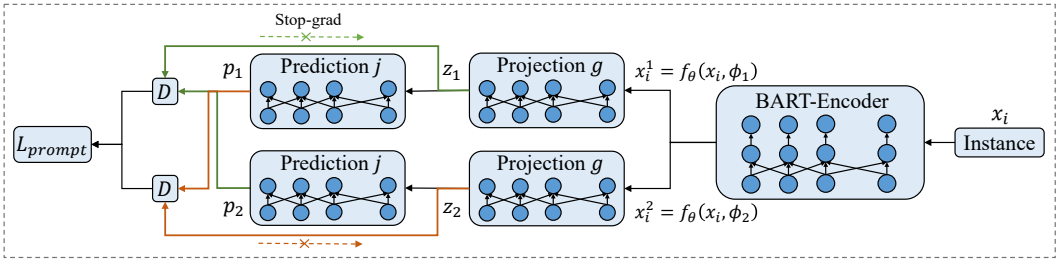

Figure 6: Siamese network-based instance contrasting.

Hyperparameters	RAMS	ACE	LR-KBP
The max length of sentence tokens	72	74	84
The max length of prompts tokens	20	18	19
Learning rate	0.0001	0.0001	0.0001
Dropout rate	0.3	0.3	0.3
Number of layers in the Projection MLP	3	3	3
Number of layers in the Prediction MLP	2	2	2

Table 2: The hyperparameter setting for our models on three datasets.

Causal models consist of FS-Causal (Chen et al., 2021), CausalProbe (Cao et al., 2022), and PAIE-debias (Lin et al., 2022). These models apply causal inference to deal with all biases in event detection or prompt learning. We executed CausalProbe and PAIE-debias since they did not provide results on our experimental datasets.

3.3. Hyperparameters and implementation details

We evaluate all models on "5+1-way 5-shot" and "10+1-way 10-shot" task. That is to say, we use 5+1 or 10+1 event types for validation and testing. Besides, we use Stochastic Gradient Decent (SGD) optimizer for training. In addition, the total epoch for training is 6000, and we evaluate the validation/test set every 400 epochs.

Other hyperparameter settings of our model on the three datasets are shown in Table 2.

3.4. Overall results

Table 3 presents the F-scores of development and testing sets on three datasets. From this table, we can observe that:

Our 2xInter outperforms the top prototype-based method by a margin of [2.5%-4.3%] on three datasets. This improvement is attributed to prompt learning in applying pre-trained language models to promote few-shot tasks and causal intervention in mitigating biases.

Our model surpasses prompt-based approach by a range of [3.1%-14.8%] on RAMS, ACE, and LR-KBP in the 5+1-way 5-shot setting. This result demonstrates that our generated prompts and prototype-based verbalizers are more effective than those in MetaEvent. Moreover, causal intervention in our model further benefits the performance. Lastly, MetaEvent achieves the best performance on LR-KBP in the 10+1-way 10-shot setting, as the prompts and verbalizers in the model are suitable for the specific situation.

2xInter outperforms the leading causal model by a range of [0.7%-2%]. Our 2xInter addresses both instance and prompt bias, whereas PAIE-debias only consider one confounder bias, explaining the superior performance. Secondly, 2xInter applies Double-View causal interventions, leading to more precise outcomes compared to intervention calculation in CausalProbe. Furthermore, incorporating prompt learning is crucial in 2xInter's superior performance over FS-Causal.

In summary, comprehensive results underscore the effectiveness of our 2xInter on few-shot event detection.

3.5. Discussion

Our 2xInter combines the PPII and DVCI modules. DVCI consists of two modules: Instance Intervention (II) and Prompt Intervention (PI). We designed four baselines: (1) 2xInter w/o II expresses that we ablate the II module. (2) 2xInter w/o PI denotes that we ablate the IC module. (3) 2xInter w/o DVCI means that we ablate the DVCI module. (4) 2xInter w/o DVCI-PPII indicates that we ablate the DVCI and PPII modules. After removal, we regard BERT-GCN as an encoder and prototype network as a classifier for few-shot event detection. Meanwhile, we devise the following experiments to analyze the effectiveness of these modules.

3.5.1. Ablation study

Table 4 presents the results of the ablation study. From the table, we can observe the following:

PPII enables effective interaction between input texts and prompts, and allows prototypical verbalizers to assign appropriate event types. As a result, 2xInter w/o DVCI achieves a 0.8% to 3.6% lead over 2xInter w/o DVCI-PPII.

2xInter shows a range of [1.2% to 1.7%] improvement over 2xInter w/o II, which indicates that the II

	Models	5+1-way 5-shot						10+1-way 10-shot					
		RAMS		ACE		LR-KBP		RAMS		ACE		LR-KBP	
		Dev	Test	Dev	Test	Dev	Test	Dev	Test	Dev	Test	Dev	Test
BERTMLP	Proto*	79.7	68.2	82.9	79.3	83.9	82.1	73.4	61.7	81.5	78.4	80.7	78.0
	InterIntra*	79.7	69.2	82.7	79.8	84.9	82.4	74.3	61.8	81.4	78.5	80.2	78.4
	DMB-Proto*	73.2	66.9	72.9	71.9	79.8	75.2	60.1	53.8	69.5	68.2	67.4	66.2
	ProAcT*	79.7	74.3	84.5	83.0	84.1	83.1	73.2	62.3	82.5	80.5	80.7	78.7
	Proto-tuning	82.8	75.3	87.1	85.6	92.1	89.3	74.9	64.1	84.1	82.8	89.2	85.9
	ProAcT-tuning	82.0	76.4	86.3	84.3	91.6	89.0	74.2	63.2	83.7	82.0	89.2	86.0
BERTGCN	Proto*	82.0	71.0	83.5	82.1	87.2	84.8	72.4	60.7	83.3	80.4	83.2	80.0
	InterIntra*	81.3	72.4	82.8	82.3	87.1	85.0	73.7	61.9	83.0	80.7	82.8	80.5
	DMB-Proto*	54.9	47.2	61.4	60.9	70.8	63.3	54.3	43.0	69.4	69.7	65.8	60.4
	ProAcT*	82.1	75.7	86.7	84.7	84.7	87.3	73.6	62.9	83.7	81.9	85.4	83.1
	Proto-tuning	83.6	76.4	87.7	87.5	91.5	88.4	75.3	64.8	84.9	84.5	87.9	84.1
	ProAcT-tuning	83.5	77.2	88.5	88.1	92.0	89.2	75.0	64.0	86.2	84.9	88.9	85.7
	HCL-TAT*	–	–	–	–	–	66.9	–	–	–	–	–	66.0
	KE-PN*	–	–	–	69.8	–	78.2	–	–	–	–	–	–
	P4E*	–	–	–	–	–	–	–	–	–	–	–	92.7
	MetaEvent	–	71.2	–	87.5	–	89.0	–	54.3	–	74.3	–	**95.8***
	FS-Causal	–	–	–	76.9	–	–	–	–	–	–	–	–
	PAIE-debias	85.2	78.9	**91.5**	89.9	94.4	91.0	77.5	67.5	87.3	85.7	**93.8**	88.5
	CausalProbe	85.1	78.8	88.9	88.1	94.5	91.2	77.3	67.2	86.8	85.6	92.6	88.0
	2xInter	**86.0**	**80.9**	91.1	**90.6**	**95.3**	**92.6**	**80.0**	**69.1**	**88.2**	**87.7**	93.7	90.0

Table 3: Overall results. Models with * are taken from their original paper.

module effectively alleviates the prompt bias.

2xInter outperforms 2xInter w/o PI by a range of [0.2% to 2.4%], which demonstrates that the PI module effectively mitigates the instance bias.

2xInter surpasses 2xInter w/o DVCI by a margin of [1.3% to 3.0%], signifying the effectiveness of the MVCI module in obviating both instance bias and prompt bias.

3.5.2. The effect of DVCI on robustness

To better understand the impact of DVCI on robustness, we ran 2xInter and 2xInter w/o DVCI five times on the ACE dataset, and then presented the averages and variances of the F-scores for these two methods in Table 5.

As demonstrated in Table 5, the variances of the F-scores increase when DVCI module is removed. This result signifies that we construct a suitable SCM and employ the causal intervention to eliminate the prompt and instance biases from our model, consequently improving the robustness.

3.5.3. The effect of prompt intervention on generalization

To investigate the influence of the prompt intervention, a Siamese network-based instance contrasting, on generalization, we show the F-scores for 2xInter and 2xInter w/o PI across all datasets in Fig. 7. The top bar graph displays results for the development set, and the bottom bar graph shows results for the test set.

Fig. 7 demonstrates that 2xInter outperforms 2xInter w/o PI on the test set, although they exhibit similar performance on the validation set. This

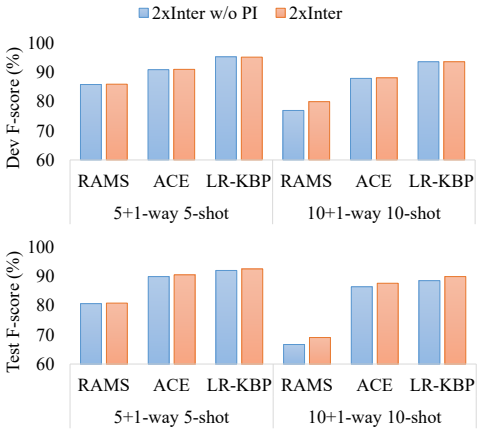

Figure 7: Two bar charts depicting F-scores of two models across all datasets.

result implies that the PI module can enhance the generalization ability, as it leverages the instance contrasting method to yield more discriminative representations.

3.5.4. The interactive method vs. the concatenating method

In our 2xInter model, instances are fed into the BART-Encoder, and the output of the encoder interacts with the prompts at the BART-Decoder. In a concatenating method, sentences and prompts are concatenated and fed into an encoder. Here, we compare the experimental performance and GPU memory usage for the interactive and concatenating methods, respectively. For the concatenating method, we utilize BERTMLP as the encoder and prototype network as the classifier. To ensure

Models	5+1-way 5-shot						10+1-way 10-shot					
	RAMS		ACE		LR-KBP		RAMS		ACE		LR-KBP	
	Dev	Test	Dev	Test	Dev	Test	Dev	Test	Dev	Test	Dev	Test
2xInter w/o DVCI-PPII	83.6	76.4	87.7	87.5	91.5	88.4	75.3	64.8	84.9	84.5	87.9	84.1
2xInter w/o DVCI	84.8	79.6	88.7	88.3	93.2	89.6	76.1	66.5	87.3	85.9	92.8	87.7
2xInter w/o PI	85.9	80.7	91.0	90.0	**95.4**	92.1	77.0	66.7	88.0	86.5	**93.7**	88.6
2xInter w/o II	**86.1**	79.7	89.5	88.9	94.8	91.2	78.6	67.3	87.0	86.4	92.5	88.8
2xInter	86.0	**80.9**	**91.1**	**90.6**	95.3	**92.6**	**80.0**	**69.1**	**88.2**	**87.7**	**93.7**	**90.0**

Table 4: Ablation study of our proposed components on "5+1-way 5-shot" and "10+1-way 10-shot" settings on RAMS, ACE, and LR-KBP datasets.

Models	dev	test
2xInter w/o DVCI	89.75 ± 1.11	88.72 ± 0.73
2xInter	91.25 ± 0.13	90.47 ± 0.32

Table 5: The averages and variances of F-scores on ACE dataset for 5+1-way 5-shot setting.

Models	Dev (F-score)	Test (F-score)	GPU memory (MB)
PPII	88.7	88.3	16487
Concatenating	85.1	84.3	22933

Table 6: The comparison of interactive and concatenating methods on ACE dataset for 5+1-way 5-shot setting.

fairness, both methods share the same hardware platform[4] and version of the Torch package[5]. The results are presented in Table 6.

From Table 6, we can observe that the concatenating method obtains lower performance and consumes more GPU memory. This result suggests that the interactive approach integrates the prompts better and reduces the GPU memory usage.

4. Related works

4.1. Prototype-based methods for few-shot event detection

Snell et al. (Snell et al., 2017) utilized a prototype network to implement few-shot classification. After that, Lai et al. (Lai et al., 2020a) introduced Few-Shot Event Detection (FSED) and presented an Intra-cluster matching and Inter-cluster information (InterIntra) method. Following this, Lai et al. (Lai et al., 2021) proposed the Prototype Representations Across Few-Shot Tasks (ProAct) method to obviate sample bias and abnormal samples. These methods solely classify event types for FSED. Deng et al. (Deng et al., 2020) regarded FSED as a two-stage task and devised a Dynamic-Memory-Based Prototypical Network (DMB-PN). However, DMB-PN struggles with error propagation. Zhang et al. (Zhang et al., 2022) proposed a Hybrid Contrastive Learning method with a Task-Adaptive Threshold (HCL-TAT) , which jointly conducts identification and classification, avoiding this issue. Zhao et al. (Zhao et al., 2022) also considered joint learning and presented a Knowledge-Enhanced self-supervised Prototypical Network (KE-PN).

Our method aligns with the works of Lai et al. (Lai et al., 2021), which focuses on event classification.

4.2. Prompt-based methods for few-shot event detection

Li et al. (Li et al., 2022) proposed the P4E model, which splits event detection into identification and location. Unlike this two-stage method, Yue et al. (Yue et al., 2023) devised a joint learning method called MataEvent, which employs a cloze-based prompt and a trigger-aware soft verbalizer. Furthermore, Song et al. (Song et al., 2023a) only focused on event classification and proposed a taxonomy-aware prompt learning framework (TaxonPrompt). In that same year, Song et al. (Song et al., 2023b) devised a knowledgeable augmented-trigger prompt FSEC framework (AugPrompt) for event classification.

We use T5-generated prompts and prototype-based verbalizers for FSED. Besides, we identify and eliminate the instance and prompt biases in prompt-based methods from a causal perspective.

4.3. Causal intervention in event detection and prompt learning

Chen et al. (Chen et al., 2021) solved the trigger curse in FSED from a causal view. Moreover, Cao et al. (Cao et al., 2022) utilized sampling-based approximation to implement causal intervention, which is used to mitigate three biases in prompt-based probing. Besides, Lin et al. (Lin et al., 2022) employed causal intervention to alleviate the confounder bias in prompt-based event argument extraction.

In our work, we propose a Double-View Causal intervention (DVCI) module to eliminate instance and prompt biases in FSED. Our DVCI uses a generation-based prompt adjustment and a Siamese network-based instance contrasting to facilitate calculation.

[4]GPU: NVIDIA GeForce RTX 3080(10 GB) * 1

[5]Torch: 1.7.0 + cu110

5. Conclusion

This paper proposes a novel FSED model called 2xInter. In the 2xInter, we first propose a PPII module to efficiently integrate sentences and prompts and directly build verbalizers from class prototypes. Next, we establish an SCM to analyze all biases in PPII and devise a DVCI module to eliminate these biases, thereby improving both performance and robustness for FSED. Moreover, the DVCI module comprises a generation-based prompt adjustment module and a Siamese network-based instance contrasting module to facilitate causal intervention. Finally, the experimental results verify the effectiveness of 2xInter.

6. Acknowledgements

This work is supported by grant from the National Natural Science Foundation of China (No. 62076048), the Science and Technology Innovation Foundation of Dalian (2020JJ26GX035).

7. Bibliographical References

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250.

Boxi Cao, Hongyu Lin, Xianpei Han, Fangchao Liu, and Le Sun. 2022. Can prompt probe pretrained language models? understanding the invisible risks from a causal view. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5796–5808.

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.

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.

Seth Ebner, Patrick Xia, Ryan Culkin, Kyle Rawlins, and Benjamin Van Durme. 2020. Multi-sentence argument linking. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8057–8077.

Viet Lai, Franck Dernoncourt, and Thien Huu Nguyen. 2021. Learning prototype representations across few-shot tasks for event detection. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5270–5277.

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

Viet Dac Lai, Tuan Ngo Nguyen, and Thien Huu Nguyen. 2020b. Event detection: Gate diversity and syntactic importance scores for graph convolution neural networks. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5405–5411.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880.

Sha Li, Liyuan Liu, Yiqing Xie, Heng Ji, and Jiawei Han. 2022. P4e: Few-shot event detection as prompt-guided identification and localization. *arXiv preprint arXiv:2202.07615*.

Jiaju Lin, Jie Zhou, and Qin Chen. 2022. Causal intervention-based prompt debiasing for event argument extraction. *arXiv preprint arXiv:2210.01561*.

David Milne and Ian H Witten. 2008. Learning to link with wikipedia. In *Proceedings of the 17th ACM conference on Information and knowledge management*, pages 509–518.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551.

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.

Chengyu Song, Fei Cai, Mengru Wang, Jianming Zheng, and Taihua Shao. 2023a. Taxonprompt: Taxonomy-aware curriculum prompt learning for few-shot event classification. *Knowledge-Based Systems*, 264:110290.

Chengyu Song, Fei Cai, Jianming Zheng, Xiang Zhao, and Taihua Shao. 2023b. Augprompt: Knowledgeable augmented-trigger prompt for few-shot event classification. *Information Processing & Management*, 60(4):103153.

Christopher Walker, Stephanie Strassel, Julie Medero, and Kazuaki Maeda. 2005. Ace 2005 multilingual training corpus-linguistic data consortium. *URL: https://catalog. ldc. upenn. edu/LDC2006T06*.

Sen Yang, Dawei Feng, Linbo Qiao, Zhigang Kan, and Dongsheng Li. 2019. Exploring pre-trained language models for event extraction and generation. In *Proceedings of the 57th annual meeting of the association for computational linguistics*, pages 5284–5294.

Zhenrui Yue, Huimin Zeng, Mengfei Lan, Heng Ji, and Dong Wang. 2023. Zero-and few-shot event detection via prompt-based meta learning. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7928–7943.

Ruihan Zhang, Wei Wei, Xian-Ling Mao, Rui Fang, and Dangyang Chen. 2022. Hcl-tat: A hybrid contrastive learning method for few-shot event detection with task-adaptive threshold. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 1808–1819.

Kailin Zhao, Xiaolong Jin, Long Bai, Jiafeng Guo, and Xueqi Cheng. 2022. Knowledge-enhanced self-supervised prototypical network for few-shot event detection. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 6266–6275.