Event Detection via Derangement Reading Comprehension

Anonymous ACL submission

Abstract

Event detection (ED), aiming to detect events from texts and categorize them, is vital to understanding the actual happenings in real life. Recently, ED without triggers has been proposed and gained benefits since it relieves the tedious effort of data labeling. However, it still suffers from several formidable challenges: multi-label, insufficient clues, and imbalanced event types. We, therefore, propose a novel Derangement mechanism on a machine Reading Comprehension (DRC) framework to tackle the above challenges. More specially, we treat the input text as *Context* and concatenate it with all event types that are deemed as *Answers* with an omitted default question. Thus, by appending input text and event types simultaneously, we can facilitate the power of self-attention in pre-trained language models, e.g., BERT, to absorb the semantic relation among them. Moreover, we design a simple yet effective *derangement* mechanism to relieve the imbalanced training. By introducing such perturbation mainly on major events, we can prohibit major events from excessive learning or implicitly under-sample the instances of the major events. This yields a more balanced training to resolve the imbalanced learning issue. The empirical results show that: (1) our proposed framework attains state-of-the-art performance over previous competitive models, and (2) by-product, our model can signify the connection of triggers and arguments to events for further analysis.

1 Introduction

The task of event detection (ED), aiming to spot the appearance of predefined event types from texts and classify them, is vital to understanding the actual happenings in real life (Edouard, 2017; Saeed et al., 2019). Take an example from ACE (Automatic Context Extraction):

> **S:** And they sent him to Baghdad and killed.

This sentence consists of two events, *Transport* and *Die*. A desired ED system should correctly identify these two events simultaneously. At first glance, this task can be arduous and challenging because event types implicitly exist in sentences.

In the literature, researchers usually tackle this problem via a two-stage trigger-based framework. That is, triggers (i.e., words or phrases providing the *most clear* indication of an event occurrence) are first identified and then events are recognized accordingly (Ahn, 2006; Li et al., 2013; Chen et al., 2015). For example, in the above example, "sent" and "killed" are the triggers for *Transport* and *Die*, respectively. Following this line, various methods have been proposed, including such as extracting syntactic, discourse, and other hand-engineered features as inputs for structured prediction (Li et al., 2013; Yang and Mitchell, 2016; Liu et al., 2018b) and neural architecture for joint tasks optimization (Nguyen et al., 2016; Nguyen and Nguyen, 2019; Wadden et al., 2019; Liu et al., 2018a). However, trigger identification is an intermediate step for event detection and requires demanding effort on annotation. After discovering triggers are nonessential to event detection, trigger-free methods, e.g., the Type-aware Bias Neural Network with Attention Mechanisms (TBNNAM) (Liu et al., 2019), have been proposed.

In this paper, we focus on event detection without triggers due to the light need of data labeling. We aim at tackling the following formidable challenges: (1) **Multi-label issue:** Each input sentence may hold zero or multiple events, which can be formulated into a challenging machine learning task, or multi-label classification task. (2) **Insufficient clues:** Triggers are of significance to attain good performance on event detection (Zhang et al., 2020; Ebner et al., 2020). Without explicitly including triggers, we may lack sufficient clues to identify the event types and need to seek alternatives to shed light on the correlation between words and the event types. (3) **Imbalanced events distribution:** As shown in Fig. 2, events may follow the Matthew effect. That is, some events dominate the data while others contain only several instances. The imbalanced event distribution brings significant obstacles to detect minor events.

Hence, we propose a Derangement mechanism on a machine Reading Comprehension (DRC) framework to tackle the challenges. Figure 1 illustrates our proposed framework with three main modules: the RC encoder, the event derangement module (EDM), and the multi-label classifier. In the RC encoder, the input sentence, deemed as "Context", and all event tokens, appended as "Answers", are fed into BERT (Devlin et al., 2019) together. Such design allows the model to observe all available information without the need of explicitly indicating triggers and enables the model to automatically learn helpful semantic relations between input texts and event tokens through the self-attention mechanism of Transformer (Vaswani et al., 2017). During training, the EDM is activated only when the grand-truth event is a major event with a certain probability. By perturbing the order of other major event tokens, the model can prevent excessively updating the instances of major events, which implicitly under-samples the training instances of the major events and yields a more balanced training to resolve the imbalanced learning issue. Finally, the learned contextual representations of event tokens are fed into a multi-label classifier to produce the probabilities of the input text to each event type.

In summary, the contribution of our work is threefold: (1) We propose a competitive paradigm to an important task, namely multi-label event detection without triggers. Through a simplified machine reading comprehension framework, we can directly capture the semantic relation between input texts and event types without explicitly including triggers. (2) During training, we implement a simple yet effective mechanism, i.e., the derangement mechanism, to overcome the imbalanced training issue. By perturbing the order of major event tokens, we implicitly under-samples the training instances from the major events and fulfill a more balanced training. (3) We report that our proposal achieves the state-of-the-art performance at event detection on the public benchmark dataset. The results also exhibit the potential of our proposal to link the triggers to the corresponding events and simultaneously signify the necessary arguments.

2 Related Work

Event Extraction (EE) A major stream of approaches focus on event extraction to identify both triggers and arguments, which can be categorized as trigger-based approaches. For example, in (Li et al., 2013), structured Perceptron has been exploited on hand-made features to identify triggers and arguments. In (Nguyen et al., 2016), triggers and arguments are jointly identified by utilizing bidirectional recurrent neural networks. In (Zhang et al., 2019), reinforcement learning is deployed with generative adversarial networks for entity and event extraction. Furthermore, witnessing the success of attention mechanism, many approaches have tried to integrate attention into the proposed models. For example, in (Liu et al., 2018b), syntactic contextual representations are learned by graph convolutional networks to extract triggers by self-attention. In (Wadden et al., 2019), a BERT-based model was proposed to multi-task learning for named-entity recognition, relation extraction, and event extraction. Another stream of trigger-based approaches formulate the EE task as a machine reading comprehension or question answering task (Du and Cardie, 2020; Liu et al., 2020). For example, in (Du and Cardie, 2020), a predefined question template concatenating with the input sentence is fed into BERT to identify the corresponding triggers and arguments. In (Liu et al., 2020), similar framework is proposed with different template design. However, this kind of implementation has to search optimal results from multiple predefined templates during inference.

Event Detection without Triggers The above trigger-based methods heavily rely on manually annotated triggers, which is time-consuming. Therefore, researchers have explored alternative trigger-free methods for event detection. The Type-aware Bias Neural Network with Attention Mechanisms (TBNNAM) (Liu et al., 2019) has been proposed to utilize the attention mechanism to incorporate information of event types to input sentences. One shortcoming is that it turns a multi-label event detection into binary classification on each event, which can be tedious and inefficient during inference.

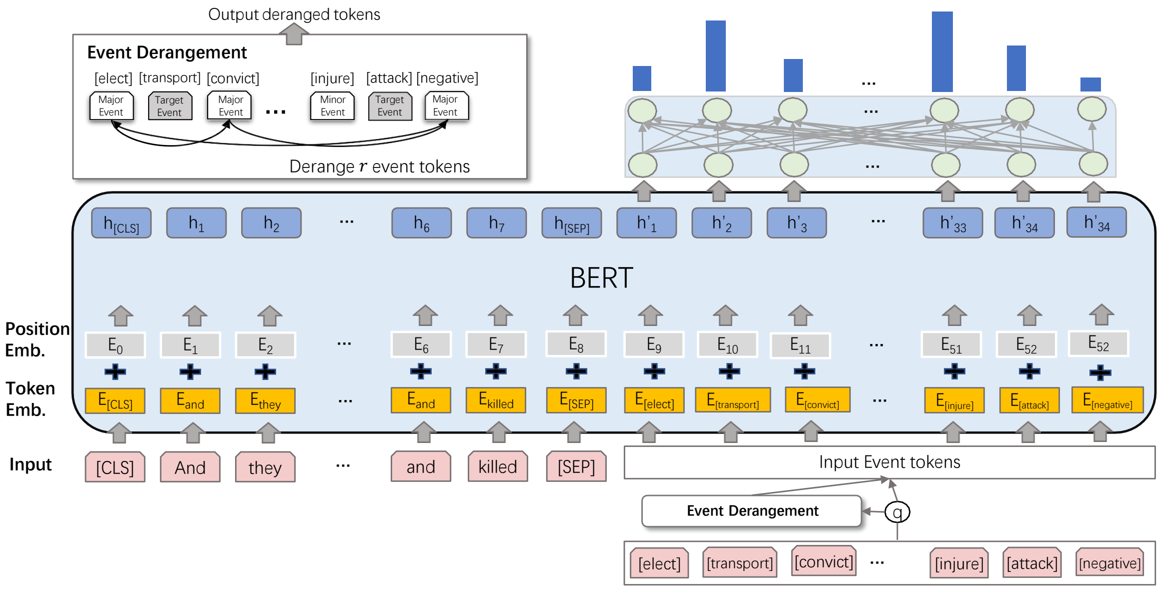


Figure 1: Our proposed DRC is on top of BERT. It consists of three main modules: RC encoder, the event derangement module (EDM), and the multi-label classifier. The EDM is amplified in the upper-left corner for better illustration; see more description in the main text.

3 Methodology

3.1 Task Definition

Following (Ahn, 2006; Ji and Grishman, 2008; Liu et al., 2019), we are given a set of training data, $\{(x_i, y_i)\}_{i=1}^{N}$, where N is the number of sentence-event pairs. $x_i = w_{i1}w_{i2}\ldots w_{i|x_i|}$ is the i-th sentence with $|x_i|$ tokens and $y_i \subseteq \mathcal{S}$ is an event set, which records all related event(s). $\mathcal{S} = \{e_1, e_2, \ldots, e_n\}$ consists of all n events, including an additional "negative" event meaning that sentences do not contain any events. Our goal is to train a model to detect the corresponding event type(s) as accurate as possible given an input sentence. This can be formulated as the multi-label classification task in machine learning. Our tasks lie in (1) how to learn more precise representations to embed the semantic information between texts and event types? (2) How to deliver the multi-task classification task effectively?

Major Events vs. Minor Events Imbalanced event distribution is a major issue in our setting. Traditionally, Imbalance Ratio (IR) (Galar et al., 2012) is a typical metric to estimate the imbalance of the data. However, IR provides little distribution information about the middle classes (Ortigosa-Hernández et al., 2017). To articulate the differences of major events and minor events, we borrow its definition in (Dong et al., 2018) to distinguish them. We first sort all event types in descending order with respect to the number of instances in each class and obtain the sorted sequence:

$$S_{\text{SA}} = e_1 \ldots e_n, \quad \text{where } |e_i| \geq |e_{i+1}|. \tag{1}$$

Here, e_i denotes the i-th event type with $|e_i|$ instances.

Then, we define the set of major events as the top-k elements in S_{SA} while the remaining elements as the minor events:

$$E_{\text{Major}} = \{e_i \mid i = 1, 2, \ldots k\}, \tag{2}$$

$$E_{\text{Minor}} = \{e_i \mid i = k+1, \ldots n\}, \tag{3}$$

where k is determined by a hyperparamter of α by rounding to the nearest integer if it is a float. Here, α indicates the percentage of the major events in all N sentence-event pairs:

$$\alpha * N = \sum_{i=1}^{k} |e_i|. \tag{4}$$

Usually, α is simply set to 0.5 as (Dong et al., 2018).

3.2 Our Proposal

Figure 1 outlines the overall structure of our proposed DRC, which consists of three main modules: the RC encoder, the multi-label classifier, and the event derangement module (EDM) for training.

RC Encoder Our proposal is based on BERT due to its power in learning the contextual representation in the sequence of tokens (Devlin et al., 2019). We present a simplified machine reading comprehension (MRC) framework:

[CLS] Context [SEP] Answers

where Context is the input sentence and Answers sequence all the event types. This setup is close to MRC with the multiple choices option. That is, it views the input sentence as Context and event types as the multiple choices (or Answers) with an omitted default question: "What is the event type/what are the event types in the Context?". With both input texts and event tokens for the input of BERT, we can utilize BERT to automatically capture the relation between input texts and event types without explicitly indicating the triggers.

Algorithm 1 Event Derangement

Require: Input sentence x; The initial event sequence S_{init}; The descending sorted sequence of all event types S_{SA}; Possibility q; Number r

Ensure: Deranged sequence of event tokens S_{O}

Initialize E_{GT} as the set of the ground truth event types implied by x
Initialize E_{D} with r events that are not in E_{GT} from the beginning sequence of S_{SA}
Initialize $E_{tmp} = \emptyset$ # a helper set to record the selected event types during derangement
Initialize $S_{\text{O}} = []$
Generate $rand$ uniformly from [0, 1]
if $E_{\text{GT}} \cap E_{\text{Major}} \neq \emptyset$ and $rand < q$ **then**
  for e_{curr} in S_{init} **do**
    if e_{curr} in E_{D} **then**
      Randomly select e from E_{D} and $e \neq e_{curr}$ and $e \notin E_{tmp}$
      Append e to S_{O}
      Add e to E_{tmp}
    else
      Append e_{curr} to S_{O}
    end if
  end for
else
  $S_{\text{O}} = S_{\text{init}}$
end if
Return S_{O}

In the implementation, given a training set, we first generate a random event order index $I_{\text{init}} = s_1 \ldots s_n$, which is a permutation of $\{1, \ldots, n\}$, and obtain its initial sequence of event tokens $S_{\text{init}} = e_{s_1} \ldots e_{s_n}$. Without specifying, the event sequence is kept fixed for both training and testing. Hence, given a sentence $x = w_1 \ldots w_{|x|}$, we obtain

$$\text{Input} = [\text{CLS}]\, w_1 \ \ldots \ w_{|x|} \ [\text{SEP}]\, e_{s_1} \ \ldots \ e_{s_n}. \quad (5)$$

To avoid word-piece segmentation, we employ a square bracket around an event type, e.g., the event token of *Transport* is converted to "[Transport]". This allows us to learn more precise event token representations and yield better performance; see more discussion in Appendix A.1. Next, we apply position embeddings based on the order of event tokens following the standard setup of BERT, although linguistically, there should be no sequential difference to event types.

After that, we learn the hidden representations:

$$\begin{aligned} h_{[\text{CLS}]}, h_1^w, \ldots, h_{|x|}^w, h_{[\text{SEP}]}, h_1^e, \ldots, h_n^e \\ = \text{BERT}(\text{Input}), \end{aligned} \quad (6)$$

where h_i^w is the hidden state of the i-th input token and h_i^e is the hidden state of the corresponding event type, namely e_{s_i}.

Multi-label Classifier After learning the contextualized representations of the Input, we turn to construct the multi-label classifier. Traditional methods usually apply a Multi-Layer Perception (MLP) on the [CLS] token to yield the classifier. Differently, we feed all hidden states of event types to a MLP for the classification due to the supportive evaluation in Appendix A.2. Hence, given an input sentence x, we compute the predicted probability for the corresponding events by

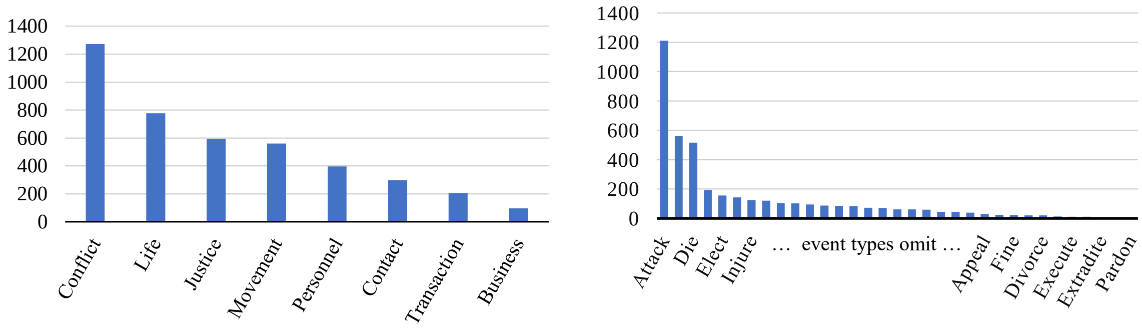
$$\hat{p} = \sigma\left(\text{MLP}\left(h_1^e, \ldots, h_n^e\right)\right). \quad (7)$$

Since $\hat{p}$ is normalized to the range of 0 and 1, for simplicity, we follow (Liu et al., 2019) to determine the event labels when $\hat{p} \geq 0.5$.

Our model can then be trained by minimizing the following loss:

$$\mathcal{L} \propto -\sum_{i=1}^{N}\sum_{j=1}^{n}(p_{ij}\log(\hat{p}_{ij}) + (1-p_{ij})\log(1-\hat{p}_{ij})) \quad (8)$$

where $p_{ij} = 1$ represents the corresponding event for the i-th input text. Different from (Liu et al., 2019) that converts the multi-label classification task into a series of binary classification tasks, our proposal can outputs all event type(s) simultaneously.



(a) Event main types

(b) Event subtypes

Figure 2: The distribution of event main types and event subtypes on the ACE2005 training data.

EDM The event derangement module is the key to resolve the imbalanced learning issue. *Derangement* is a term in combinatorics, where a permutation of the elements in a set makes no element appearance in its original position. In the implementation, when the target (i.e., the ground-truth) events are major events, we deliver the derangement procedure with probability q; see line 6 of Algo. 1. Moreover, only events in E_{D} and not preselected are selected from derangement; see line 9 of Algo. 1. It is noted that E_{D} is r events excluding the target events in E_{GT} from the beginning of S_{SA} (i.e., usually the deranged events are major events; see the definition in line 2 of Algo. 1). The parameter r allows us to determine the number of the events for derangement.

The underlying effect of derangement is to compress the learning of major events, which is close to under-sampling the training instances from major events. This can yield a more balanced training process and resolve the imbalanced learning issue. We provide more supporting results and explanations in Sec. 5.2.

4 Experiments

We present the experimental setups and overall performance in the following.

4.1 Experimental Setups

Dataset and Evaluation We conducted experiments on the ACE2005 English corpus. The ACE2005 corpus consists of 8 event main types and 33 subtypes. As shown in Fig. 2, the corpus follows the imbalanced event distribution and is more imbalanced (IR$\approx$605.5) for the event subtypes than that (IR$\approx$13.1) in the event main types. For example, the types of *Attack*, *Transport*, and *Die* account for over half of the total training data. For fair comparison, we follow the evaluation of (Li et al., 2013; Liu et al., 2019, 2020), i.e., randomly selecting 30 articles from different genres as the validation set, subsequently delivering a blind test on a separate set of 40 ACE2005 newswire documents, and using the remaining 529 articles as the training set. The standard metrics: Precision (P), Recall (R), and F1 scores (F1), are applied to evaluate the model performance.

Implementation Details Our implementation is in PyTorch[1]. The *bert-base-uncased* from Hugging Face (Wolf et al., 2019) is adopted as the backbone model. The MLP consists of two layers with the hidden size being 768 and yields an output of 34 dimension to predict the probability of the input sentence assigned to the corresponding 34 classes. We follow (Dong et al., 2018) to set α as 0.5 and round k to the nearest integer based on the calculation by Eq. (4). In EDM, the derangement probability q is set to 0.2 and r is 24 from empirical selection. The batch size is 8. The dropout rate is 0.1. ADAM is the optimizer (Kingma and Ba, 2015) with a learning rate of 2×10^{-5}. We train our models for 10 epochs to give the best performance. All experiments are conducted on an NVIDIA A100 GPU in around 1.5 hours.

4.2 Overall Performance

We compare our proposed BERT_RC and BERT_DRC with several competitive baselines: **TBNNAM** (Liu et al., 2019): an LSTM model detecting events without triggers, and BERT-based models for both trigger detection and event

[1] https://www.dropbox.com/s/4h4p0dl26jha7q6/DRC.zip?dl=0

Methods	Subtypes (%)			Main (%)		
	P	**R**	**F1**	**P**	**R**	**F1**
TBNNAM (Liu et al., 2019)	76.2	64.5	69.9	-	-	-
DYGIE++, BERT + LSTM (Wadden et al., 2019)	-	-	68.9	-	-	-
DYGIE++, BERT Finetune (Wadden et al., 2019)	-	-	69.7	-	-	-
BERT_RC_Trigger (Du and Cardie, 2020)	71.7	73.7	72.3	-	-	-
DMBERT (Wang et al., 2019)	77.6	71.8	74.6	-	-	-
RCEE_ER (Liu et al., 2020)	75.6	74.2	74.9	-	-	-
DMBERT + Boot (Wang et al., 2019)	77.9	72.5	75.1	-	-	-
BERT Finetune	72.8	68.7	70.7	78.0	70.8	74.2
Our BERT_RC	76.9	72.3	74.7	78.9	75.4	77.1
Our BERT_DRC	**79.5**	**76.8**	**78.1**	**78.7**	**79.0**	**78.9**

Table 1: Event detection results on both the event subtypes and event main types of the ACE2005 corpus.

detection: **DYGIE++** (Wadden et al., 2019): a BERT-based framework modeling text spans; **BERT_RC_Trigger** (Du and Cardie, 2020) and **RCEE_ER** (Liu et al., 2020): both BERT-based models converting event extraction as an MRC task; **DMBERT** (Wang et al., 2019): a BERT-based model leveraging adversarial training for weakly supervised events, where DMBERT Boot stands for bootstrapped DMBERT.

Table 1 reports the overall performance on the ACE2005 corpus. It shows that (1) previous models only evaluate the performance on the event subtypes. Although our proposed BERT_RC does not access to the triggers, it attains significant better performance than TBNNAM, DYGIE++, and BERT_RC_Trigger. Its performance is also competitive to DMBERT and RCEE_ER, with 74.7% F1 score, only 0.4% less F1 score than that in the best baseline, DMBERT Boot. The result shows that our proposed RC framework is effective to learn the semantic information between given texts and event types. (2) After introducing the derangement mechanism, our proposed BERT_DRC can significantly outperform all compared methods in all three metrics. Especially, it attains 3.0% more F1 score than the best baseline. (3) To verify the generalization of our proposal, we also conduct experiments to evaluate the performance on event main types. The setting of the model parameter is the same as that on the event subtypes, except $r = 3$ for DRC. The results show that our proposed BERT_RC and BERT_DRC gain further improvement, i.e., 2.9% and 4.7% F1 score over the finetuned BERT, respectively. The results show the consistence of our proposal and it seems that BERT_DRC can attain better performance when the dataset (the event subtypes) is more imbalanced; see more supporting results in Appendix A.3.

	P	**R**	**F1**
BERT_RC_Same	75.7	71.6	73.6
BERT_RC	76.9	72.3	74.7
BERT_RC_Shuffle_Test	18.2	9.2	12.2
BERT_DRC_Shuffle_Test	66.0	45.1	53.6
BERT_DRC	**79.5**	**76.8**	**78.1**

Table 2: Evaluation results for event orders.

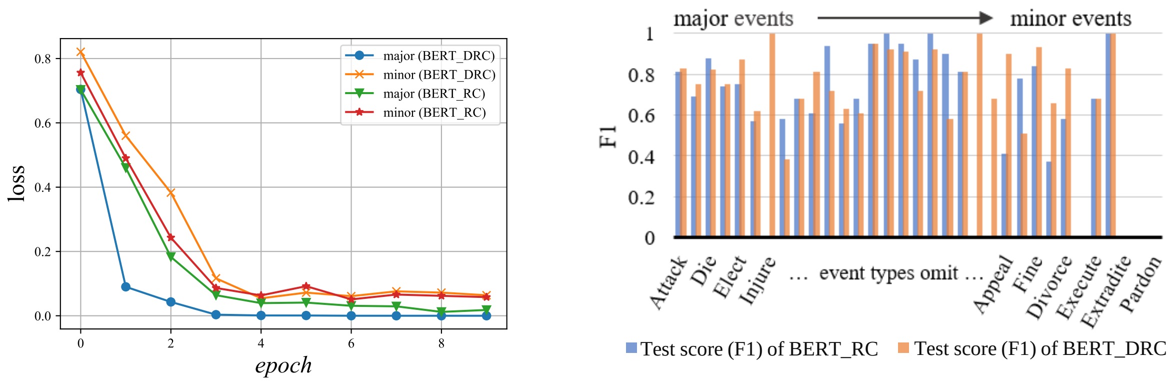
5 More Analysis

We try to discover the underlying mechanism of our proposed derangement and provide more analysis on our proposal.

5.1 Effect of Event Orders

Table 2 reports the effect of the event orders in different cases. The first two rows record the results of BERT_RC_Same and BERT_RC, where BERT_RC_Same applies the same position embeddeding to all event types to eliminate the difference in event orders. On the contrary, BERT_RC applies varied position embeddeding to each event type. The results show that by leveraging the event order, BERT_RC gains around 1% improvement on the F1 score.

We further show that our BERT_RC is order-sensitive. This can be verified by the results of BERT_RC and BERT_RC_Shuffle_Test. Here, BERT_RC_Shuffle_Test is trained with the same event order of BERT_RC, but tested with a shuffled event order. By confusing BERT_RC with a different event order during inference, we obtain a significant drop on the F1 score to 12.2%. This



(a) Loss curves

(b) Performance w/o derangement

Figure 3: Fig. 3(a) shows the losses of BERT_DRC and BERT_RC on major events and minor events, respectively. Fig. 3(b) shows the compared F1 score of BERT_RC (colored by blue) and BERT_DRC (colored by red) on the test set. For better visualization, we only show parts of events and set the label segmentation to 2.

further implies that our BERT_RC tends to rely on the event order to recognize the events.

Furthermore, by performing derangement on BERT_RC_Shuffle_Test, we obtain the result of BERT_DRC_Shuffle_Test. It shows that BERT_DRC_Shuffle_Test obtains much better performance than BERT_RC_Shuffle_Test. We conclude that BERT_DRC learns more semantic information than BERT_RC. In other words, the derangement mechanism can help BERT_RC to relieve the reliance of remembering event orders.

5.2 Effect of EDM

We show the effect of EDM to understand the underlying mechanism of how EDM works. Fig. 3(a) shows the losses of BERT_DRC and BERT_RC on major events and minor events, respectively. To amplify the effect, we set $q = 1.0$, an extreme case of EDM, where the event derangement is conducted on each training batch. It shows that due to the interference of the derangement, the loss of BERT_DRC on the major events drops much faster and is much smaller (close to zero) than the counterpart of BERT_RC. We conjecture that the swift convergence of BERT_DRC on major events comes from the leak of the position hint implied by derangement, because our model is order-sensitive. During derangement, the position of ground-truth (major) events is reserved while other events are deranged. Hence, this yields low loss and less gradient update on major events in BERT_DRC than that in BERT_RC. In other words, derangement prohibits major events from being excessively learned. The derangement procedure implicitly implements under-sampling the instances of major classes during training and thus fulfills a more balanced learning process. Our EDM may echo the mechanism in response to sensory deprivation (Merabet and Pascual-Leone, 2010): neurons in human brain are reorganized to functioning regions (i.e. minor events in our case), which, for instance, makes the blind have stronger hearing.

Figure 3(b) further reports the performance of each event by setting the derangement probability q to 0.2 and the size of derangement set r to 24, which achieves the best performance of BERT_DRC. Via derangement or a more balanced training, BERT_DRC attains an F1 score of 78.1%, a 3.4% improvement over BERT_RC. By examining the details, the main improvement comes from recognizing the minor events, i.e., improving the F1 score from 69.1% to 72.4%.

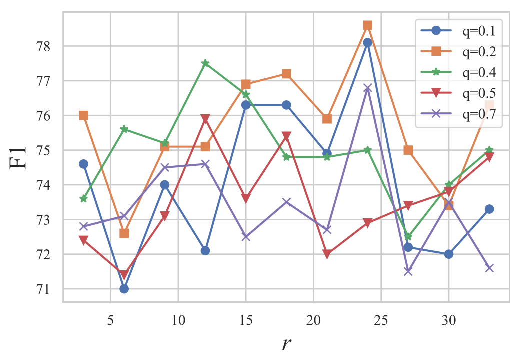


Figure 4: Effect of q and r when evaluated on ACE2005.

5.3 Effect of Hyperparameters

In this section, we test the derangement probability q and the size of the derangement set r, where q is selected from $\{0.1,\ 0.2,\ 0.4,\ 0.5,\ 0.7\}$ and r is selected from $\{3, 6, \ldots, 33\}$, i.e., equally dividing all event types into 10 buckets. We ignore larger q's because they usually fail the model on detecting major events. Figure 4 shows that the best performance is attained when $q = 0.2$ and $r = 24$. The trends also show that a smaller q can usually yield better performance than a larger one while r is selected in the range of 15 and 25 because r can determine the scale of perturbation. A smaller r may cause negligible perturbation and a larger r may affect the disturbance of the minor events. Though usually, these two parameters are data-oriented, we observe similar trend for the TAC-KBP corpus; see more results in Appendix A.5.

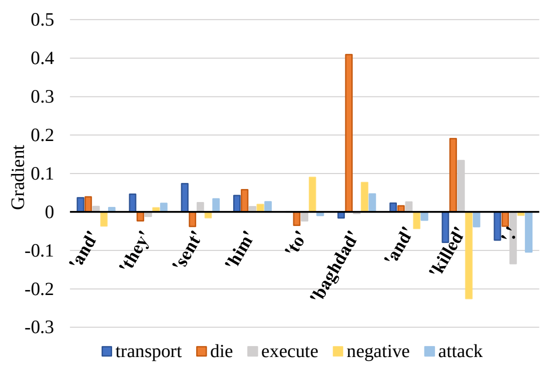


Figure 5: Gradient visualization of words in a sentence with respect to five typical event types; see more description in the main text.

5.4 Gradient Explanation

In the literature, the gradient explanation has been verified as a more stable method to explain the attention model (Adebayo et al., 2018) than the attention weights in BERT because the attention weights may be misleading (Jain and Wallace, 2019) or are not directly interpretable (Brunner et al., 2020). We then compute the gradient with respect to the input text embeddings, which quantifies the influence of changes in the tokens on the predictions. Here, we pick the example in Sec. 1 and select five typical events: "Die" and "Transport" are the target events; "Negative" and "Attack" are two common event types; and "Execute" is a minor event. Figure 5 clearly shows that

- For the event of "Die", our BERT_DRC can automatically focus on its trigger word "killed" while for the event "Transport", the trigger "sent" is also noticed by model. But for non-target events, our BERT_DRC attains low gradients on the triggers or gets high gradients on unrelated tokens, such as "to" and ".".
- More importantly, our BERT_DRC can surprisingly spot the related arguments for the events. For example, for the event of "Die", "Baghdad" yields a significant higher gradient, which corresponds to the argument of PLACE. Similarly, for the event of "Transport", "they" and "him" also yield relatively larger gradients, which exactly correspond to the argument of ARTIFACT and AGENT, respectively.

The observations shows the potential power of our proposal in not only linking triggers to the corresponding events, but also highlighting the corresponding event arguments. Our proposal can be a better tool to signify these words than traditional trigger-based methods.

6 Conclusion and Future Work

In this paper, we propose a novel Derangement Question Answering (DRC) framework on top of BERT to detect events without triggers and under the imbalanced setting. By treating the input text as a Context and directly concatenating it with all event types as Answers, we utilize the power of self-attention in BERT to absorb the semantic relation between the original input text and the event type(s). Moreover, we propose a simple yet effective derangement mechanism to relieve the imbalanced training. By delivering perturbation when the target event is a major event, we train prohibit the training and implicitly under-sample the training instance. We conduct sufficient evaluation and show that our proposed framework attains state-of-the-art performance over previous methods and can automatically link the triggers with the event types while signifying the related arguments.

Several interesting directions can be considered in the future. First, since our proposal is event-order-sensitive, it is worthy to explore how to generate an optimal initial event order. Second, the gradient explanation is effective to signify triggers and arguments. It is meaningful to merge it in our proposal to extract the key information of events. Third, it would be worthwhile to adapt our proposal to other information extraction tasks to extend its application scope.

References

Julius Adebayo, Justin Gilmer, Michael Muelly, Ian J. Goodfellow, Moritz Hardt, and Been Kim. 2018. Sanity checks for saliency maps. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 9525–9536.

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

Gino Brunner, Yang Liu, Damian Pascual, Oliver Richter, Massimiliano Ciaramita, and Roger Wattenhofer. 2020. On identifiability in transformers. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

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 of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers*, pages 167–176. The Association for Computer Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Qi Dong, Shaogang Gong, and Xiatian Zhu. 2018. Imbalanced deep learning by minority class incremental rectification. *IEEE transactions on pattern analysis and machine intelligence*, 41(6):1367–1381.

Xinya Du and Claire Cardie. 2020. Event extraction by answering (almost) natural questions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 671–683. Association for Computational Linguistics.

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, ACL 2020, Online, July 5-10, 2020*, pages 8057–8077. Association for Computational Linguistics.

Amosse Edouard. 2017. *Event detection and analysis on short text messages. (Détection d'événement et analyse des messages courts)*. Ph.D. thesis, University of Côte d'Azur, France.

Joe Ellis, Jeremy Getman, Dana Fore, Neil Kuster, Zhiyi Song, Ann Bies, and Stephanie M. Strassel. 2015. Overview of linguistic resources for the TAC KBP 2015 evaluations: Methodologies and results. In *Proceedings of the 2015 Text Analysis Conference, TAC 2015, Gaithersburg, Maryland, USA, November 16-17, 2015, 2015*. NIST.

Mikel Galar, Alberto Fernández, Edurne Barrenechea Tartas, Humberto Bustince Sola, and Francisco Herrera. 2012. A review on ensembles for the class imbalance problem: Bagging-, boosting-, and hybrid-based approaches. *IEEE Trans. Syst. Man Cybern. Part C*, 42(4):463–484.

Sarthak Jain and Byron C. Wallace. 2019. Attention is not explanation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 3543–3556. Association for Computational Linguistics.

Heng Ji and Ralph Grishman. 2008. Refining event extraction through cross-document inference. In *ACL 2008, Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics, June 15-20, 2008, Columbus, Ohio, USA*, pages 254–262. The Association for Computer Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Qi Li, Heng Ji, and Liang Huang. 2013. Joint event extraction via structured prediction with global features. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, ACL 2013, 4-9 August 2013, Sofia, Bulgaria, Volume 1: Long Papers*, pages 73–82. The Association for Computer Linguistics.

Jian Liu, Yubo Chen, Kang Liu, Wei Bi, and Xiaojiang Liu. 2020. Event extraction as machine reading comprehension. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 1641–1651. Association for Computational Linguistics.

Jian Liu, Yubo Chen, Kang Liu, and Jun Zhao. 2018a. Event detection via gated multilingual attention mechanism. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 4865–4872. AAAI Press.

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, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 735–744. Association for Computational Linguistics.

Xiao Liu, Zhunchen Luo, and Heyan Huang. 2018b. Jointly multiple events extraction via attention-based graph information aggregation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 1247–1256. Association for Computational Linguistics.

Lotfi B Merabet and Alvaro Pascual-Leone. 2010. Neural reorganization following sensory loss: the opportunity of change. *Nature Reviews Neuroscience*, 11(1):44–52.

Thien Huu Nguyen, Kyunghyun Cho, and Ralph Grishman. 2016. Joint event extraction via recurrent neural networks. In *NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016*, pages 300–309. The Association for Computational Linguistics.

Trung Minh Nguyen and Thien Huu Nguyen. 2019. One for all: Neural joint modeling of entities and events. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 6851–6858. AAAI Press.

Jonathan Ortigosa-Hernández, Inaki Inza, and Jose A Lozano. 2017. Measuring the class-imbalance extent of multi-class problems. *Pattern Recognition Letters*, 98:32–38.

Haoruo Peng, Yangqiu Song, and Dan Roth. 2016. Event detection and co-reference with minimal supervision. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 392–402. The Association for Computational Linguistics.

Zafar Saeed, Rabeeh Ayaz Abbasi, Onaiza Maqbool, Abida Sadaf, Imran Razzak, Ali Daud, Naif Radi Aljohani, and Guandong Xu. 2019. What's happening around the world? A survey and framework on event detection techniques on twitter. *J. Grid Comput.*, 17(2):279–312.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NIPS*, pages 5998–6008.

David Wadden, Ulme Wennberg, Yi Luan, and Hannaneh Hajishirzi. 2019. Entity, relation, and event extraction with contextualized span representations. 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 2019, Hong Kong, China, November 3-7, 2019*, pages 5783–5788. Association for Computational Linguistics.

Xiaozhi Wang, Xu Han, Zhiyuan Liu, Maosong Sun, and Peng Li. 2019. Adversarial training for weakly supervised event detection. 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 998–1008.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *CoRR*, abs/1910.03771.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *CoRR*, abs/1609.08144.

Bishan Yang and Tom M. Mitchell. 2016. Joint extraction of events and entities within a document context. In *NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016*, pages 289–299. The Association for Computational Linguistics.

Tongtao Zhang, Heng Ji, and Avirup Sil. 2019. Joint entity and event extraction with generative adversarial imitation learning. *Data Intell.*, 1(2):99–120.

Zhisong Zhang, Xiang Kong, Zhengzhong Liu, Xuezhe Ma, and Eduard H. Hovy. 2020. A two-step approach for implicit event argument detection. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 7479–7485. Association for Computational Linguistics.

A Appendix

We provide more analysis to support our proposal.

Conversion	P	R	F1
Original	75.2	67.6	71.7
New	77.3	68.2	72.5

Table 3: Results of different conversion ways of event tokens.

A.1 Effect of Event Tokens Conversion

There are two intuitive ways to treat the event tokens in our proposed framework. One is to treat them as old words in the BERT dictionary, so that we can initialize the event representations by utilizing BERT's pre-trained word embeddings. The other way is to treat them as new words, so that we can learn the event representations from scratch. Hence, we can directly feed the *original* event words in the DRC framework or add a square bracket around the event words to convert them into *new* words, e.g., "Transport" to "[Transport]", in the BERT dictionary.

Table 3 reports the compared results and shows that converting event types into new words can attain substantial improvement in all three metrics than treating them as the original words in BERT dictionary. We conjecture that it may arise from WordPiece (Wu et al., 2016) in BERT implementation because BERT will separate an event word into several pieces when it is relatively long. This brings the difficulty in precisely absorbing the semantic relation between the words in input texts and event types. On the contrary, when we treat an event word as a new word, BERT will deem them as a whole. Though BERT learns the event representations from scratch, it is still helpful to establish the semantic relationship between words and event types.

A.2 Inputs for the Multi-label Classifier

There are two kinds of inputs for the multi-label classifier: the representation of the [CLS] token, or the event representations. We feed these two inputs into the same MLP to predict the probability of an input sentence x to the corresponding events.

Input	P	R	F1
[CLS]	77.3	68.2	72.5
All event tokens	76.9	72.3	74.7

Table 4: Results of different inputs for the multi-label classifier.

Table 4 reports the performance of different inputs for the multi-label classifier and shows that by feeding the event representations as the input, our BERT_RC can significantly improve the performance on Recall and the F1 score with competitive Precision score than only using the representation of the [CLS] token. We conjecture that the event representations have injected more information into the multi-label classifier than only using the representation of the [CLS] token.

A.3 Limitation of EDM

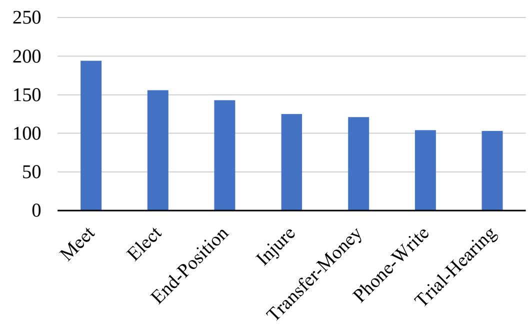


Figure 6: Data distribution of seven balanced event subtypes.

We conduct evaluation on a more balanced dataset to investigate the limitation of EDM. We first select seven relatively balance event types, yielding an imbalance ratio around 1.8, from the subtypes of the ACE2005 corpus; see the data distribution in Fig. 6. In the experiment, we set q to 0.2 and r to 6 for good performance on BERT_DRC.

Model	P	R	F1
BERT_RC	76.4	77.8	77.1
BERT_DRC	75.0	76.3	75.6

Table 5: The performance of our BERT_RC and BERT_DRC on a more balanced dataset.

Table 5 reports the comparison results of BERT_RC and BERT_DRC and shows that BERT_RC attains satisfactory results and beats BERT_DRC in all three metrics. The results imply that the derangement procedure plays an important role when the dataset is more imbalanced. When the dataset is relatively balanced, we can turn to BERT_RC and attain good performance due to the power of self-attention in BERT.

A.4 Error Analysis

We conduct error analysis on test dataset in this section. There are three main kinds of errors:

– The main error comes from event misclassification, accounting for 52.9% of the total errors. The error also includes that BERT_DRC detects more event types than the ground truth. The most event type that BERT_DRC over-predicts is the event of *Attack*. A typical example is given below:

  > **S:** The officials, who spoke on ... 26 words omitted ... on the U.S.-backed **war** resolution.

  BERT_DRC deems this sentence belonging to the event of *Attack*, where the ground truth is the event of *Meet*. This error is normal because the word “war” is a common trigger to the event of *Attack*, which yields BERT_DRC misclassifying it. In this dataset, the event of *Attack* is the most dominating event type, which makes it likely to classify the texts of other events as *Attack* when the texts hold some similar words to the triggers of *Attack*.

– The second type of errors is that BERT_DRC outputs fewer event types than the ground truth, which accounts for 28.9% of errors. The frequently missing event types are *Transfer-Money* and *Transfer-Ownership*. One typical example is

  > **S:** Until Basra, U.S. and British troops ... 6 words omitted... they **seized** nearby Umm RCsr ... 3 words omitted... **secure** key oil fields.

  BERT_DRC fails to identify the event of *Transfer-Ownership*, which is indicated by the trigger, “secure”, while recognizing the event of *Attack*, implied by the trigger if “seized”. On the one hand, the Imbalanced Ratio of *Attack* and *Transfer-Ownership* is 14.2. There are much fewer training data for BERT_DRC to learn the patterns of *Transfer-Ownership* than those of *Attack*. On the other hand, deeper semantic knowledge is needed for understanding the event of *Transfer-Ownership*, whose trigger words are more diverse and changeable. The triggers for *Transfer-Ownership* may include “sold”, “acquire”, and “bid”, etc.

– The third type of errors lies in outputting none-event sentences. When there are no event types in a sentence, BERT_DRC may fail to classify it as the type of *negative*. This is because there is no sufficient clues for BERT_DRC to learn the patterns from the type of *negative*. BERT_DRC also turns out to give low predicted probabilities on all event types.

Methods	P	R	F1
MSEP-EMD (Peng et al., 2016)	69.2	47.8	56.6
BERT Finetune	84.1	65.0	71.7
Our BERT_RC	77.4	74.8	76.1
Our BERT_DRC	79.8	75.2	77.4

Table 6: The results of our BERT_RC and BERT_DRC when evaluated on TAC-KBP 2015.

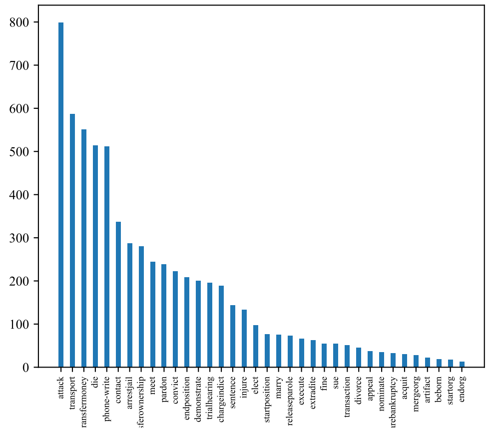
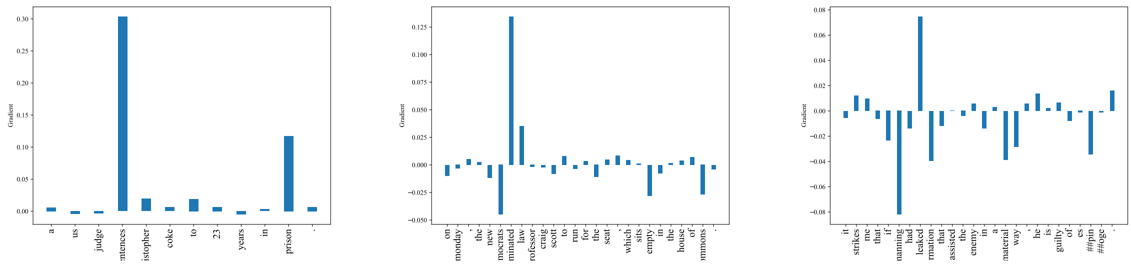


Figure 7: The event type distribution in TAC-KBP.

A.5 Evaluation on TAC-KBP

We evaluate our BERT_RC and BERT_DRC on the TAC-KBP-2015 corpus (Ellis et al., 2015). The corpus is annotated with event nuggets that fall into 38 types. We process the data following (Peng et al., 2016). The data distribution is shown in Fig. 7 and is more balanced than ACE2005 with IR $\approx$ 61.5. We implement our BERT_RC and BERT_DRC by setting the configuration in align with that on the ACE2005 dataset (see Section. 4.1). The results are shown in Table 6. It can be observed that our BERT_RC can greatly outperform the finetuned base BERT by 4.4% in F1 score. Our BERT_RC is thus a framework proven capable in both the ACE2005 English dataset and the TAC-KBP-2015 dataset. The EDM can further improve our BERT_DRC by 1.3%. But the increment is relatively smaller than that on the ACE2005 dataset, which may be caused by the different corpus and data distribution of two datasets.

Based on the TAC-KBP-2015 dataset, we investigate the effect of q and r in EDM on BERT_DRC's



(a) “sentence” as the trigger of the event: “justice:sentence”
(b) “nominated” as the trigger of the event: “transaction:nominate”
(c) “leaked” as the trigger of the event: “contact:contact”

Figure 8: Gradient visualization of words in randomly selected sentences with respect to predicted event types; see more description in the main text.

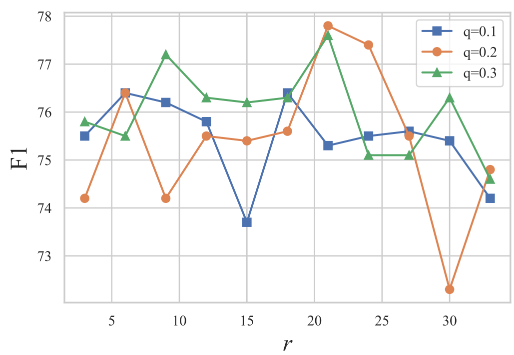


Figure 9: Effects of q and r when evaluated on TAC-KBP 2015.

performance. We select q from $\{0.1,\ 0.2,\ 0.3\}$ and r from $\{3, 6, \ldots, 33\}$.

Figure 9 shows the performance with respect to r for different q. It is shown that the best performance is attained when $q = 0.2$ and $r = 21$, reaching 77.8% for F1 score. The trends remain largely the same as those on the ACE2005 dataset. The best performances also occur when r is selected in the range of 15 and 25 because r can indicate the scale of perturbation. A smaller r may cause negligible perturbation and a larger r may affect the disturbance of the minor events.

We then conduct gradient explanation on our DRC framework as in Sec. 5.4. We randomly choose instances from the test set and visualize gradients respect to the correctly predicted event types by our BERT_DRC. As shown in Fig. 8, “nominated”, “leaked” and “sentences” are respectively triggers for those three sentences and receive significant positive gradients compared with other words. This shows that our DRC framework can automatically learn to spot triggers and relate them to event types in practice.