Augmenting Open-Domain Event Detection with Synthetic Data from GPT-2

Amir Pouran Ben Veyseh[1]✉, Minh Van Nguyen[1], Bonan Min[2], and Thien Huu Nguyen[1]

[1] Department of Computer and Information Science, University of Oregon, OR, USA
[2] Raytheon BBN Technologies, USA
{apouranb,minhnv,thien}@cs.uoregon.edu, bonan.min@raytheon.com

Abstract. Open-domain event detection (ODED) aims to identify event mentions of all possible types in text. A challenge for ODED research is the lack of large training datasets. In this work, we explore a novel method to overcome this challenge by fine-tuning the powerful pre-trained language model GPT-2 on existing datasets to automatically generate new training data for ODED. To address the noises presented in the generated data, we propose a novel teacher-student architecture where the teacher model is used to capture anchor knowledge on sentence representations and data type difference. The student model is then trained on the combination of the original and generated data and regularized to be consistent with the anchor knowledge from the teacher. We introduce novel regularization mechanism based on mutual information and optimal transport to achieve the knowledge consistency between the student and the teacher. Moreover, we propose a dynamic sample weighting technique for the generated examples based on optimal transport and data clustering. Our experiments on three benchmark datasets demonstrate the effectiveness of the propped model, yielding state-of-the-art performance for such datasets.

Keywords: Event Extraction · Natural Language Processing · Data Augmentation · GPT-2 · Teacher-Student Architecture.

1 Introduction

In Natural Language Processing (NLP), events are mentioned in text and refer to changes of states of real word entities [33]. Each event in text is associated with an event trigger, which is the main word (most often a single verb or nominalization) to evoke the event (e.g., *agree*, *death*). As such, Event Detection (ED) is one of the important tasks in Information Extraction of NLP whose goal is to identify trigger words of events in text. For instance, in the sentence "*Ames* ***recruited*** *her as an informant in 1983, then* ***married*** *her two years later*", an ED system should be able to recognize "*recruited*" and "*married*" as trigger words of two events of semantic types *Employment* and *Marriage* respectively (also called event mentions). Due to its ubiquity, detecting event mentions is an integral part of natural language understanding that can support question answering, text sumarization, and knowledge base population (among others).

There is a wealth of prior work on event detection [11, 25, 5, 24, 37, 15]. However, most of prior works on ED assumes a predefined and small set of event types for some specific domain that prevents the extraction of open-domain events (i.e., of all possible types). This work focuses on open-domain ED (ODED) to fill this gap. One hurdle for developing effective ODED models is the lack of large training datasets. To obtain more training signals/data for ED, prior works have resorted to unsupervised [9, 40], distantly supervised [12, 3] or domain adaptation [22] methods. The common characteristics of these methods involves the requirement of large-scale (though unlabeled) texts replete with event mentions. In this work, we propose a novel method to generate additional training data for ODED solely based on the pre-trained language model GPT-2 [32]. In particular, given the training data in an existing ODED dataset, we fine-tune GPT-2 to automatically generate sentences that are annotated with open-domain event mentions. The generated data will then be combined with the original training data to train models for ODED. To our knowledge, this is the first work that leverages GPT-2 for data generation/augmentation for ED.

How can we efficiently combine the generated sentences with the original data? The answer for this question is important as the generated samples might be noisy/erroneous and simply concatenating them with the original training data to train models might even hurt the performance. To address this issue, prior work on automatic data generation with GPT-2 [2, 39] has explored sample filtering techniques to remove noisy samples from the generated data. This is done once before the model training and the remaining generated data is added into the original data to train models. However, the noisiness criteria for sample selection in such prior work is solely determined by fixed heuristics that might not be optimal. To better handle the noisy data from GPT-2, we propose to instead keep all the generated samples and devise mechanisms to directly address noisy examples during the training process of the models for ODED. As such, we argue that noisy samples would cause the representations/knowledge learned via the combination of synthetic and original data to diverge significantly from those learned solely via the original training data. To minimize the effect of noisy samples, we can thus use the knowledge in the original data as an anchor and enforce the induced knowledge from the synthetic and original data to be consistent/close to those anchor. In this way, the models can still benefit from the new information presented in the synthetic data while avoiding its inherent noises via the consistency constraint.

To this end, the teacher-student architecture is employed to achieve the knowledge consistency. Here, the teacher model is first trained on the original training data for ODED to induce anchor knowledge. The student will be trained afterward from both generated and original data. Two learning principles are introduced to accommodate consistency between the knowledge from the student and the anchor knowledge: (1) Representations obtained by the teacher and the student on the same sentences should be compatible with each other. As such, we propose to increase the mutual information between the teacher and student networks in the learning process; (2) Both the teacher and the student should discern the same level of difference between the original and the generated data. For this principle, we propose two aspects to assess the distance between these two types of data for difference consistency enforcement, i.e., representations and event-containing likelihoods for sentences. In particular, instead of only relying on the similarity of the induced representations, our distance function for two sentence examples in ODED also consider the similar likelihood for the sentences to contain event mentions (i.e., event-containing sentences are more similar in ODED). Accordingly, our model leverages Optimal Transport, a method to find the cheapest transformation between two data distributions, as a natural solution to simultaneously incorporate the two aspects into the distance function for synthetic and original data for ODED.

Finally, to further regulate the synthetic data in the training process, we seek to assign weights to the generated samples in the training loss function to control their contribution for ODED. In particular, we compute the weight of each synthetic data point based on its diversity w.r.t. the other generated samples and its novelty w.r.t. the original training data. Our model features dynamic updates of the data point weights after each training epoch to better reflect the training process. Extensive experiments on three benchmark ODED datasets reveal the effectiveness of the proposed model and lead to state-of-the-art performance for all the datasets.

2 Model

We formulate ODED as a sequence labeling task. Given a sentence $S = [w_1, w_2, \ldots, w_N]$, the goal is to make a prediction for each word w_i, indicating whether it is triggering an event or not (i.e., a binary classification for each word). Our proposed approach for ODED consists of two major stages: (1) Data Generation and (2) Task Modeling.

2.1 Data Generation

Our approach is motivated by GPT-2, a transformer-based language model [32] that has shown impressive capability to generate fluent and meaningful text. As such, we aim to utilize GPT-2 to automatically generate training data for ODED[3] and combine it with existing datasets to train better ODED models. To this end, given a training dataset for ODED, we first attempt to fine-tune the pre-trained GPT-2 model on this dataset so the knowledge about open-domain event mentions can be injected into the language model. The expectation is to encourage GPT-2 to produce synthetic sentences marked with event mentions afterward.

[3] We use the small version of GPT-2 in this work.

Formally, for each sentence $S = [w_1, w_2, \dots, w_N]$ in the original training dataset $\mathcal{O}$, we first insert two special tokens TRG_s and TRG_e right before and after (respectively) each event trigger word in S. For instance, assuming w_t is the only event trigger in S, the resulting sentence S' after insertion would be: $S' = [w_1, \dots, TRG_s, w_t, TRG_e, \dots, w_N]$. Afterward, we fine-tune the pre-trained GPT-2 model on the newly generated sentences S' in an auto-regressive fashion (i.e., predicting the next token in S'). Finally, the fine-tuned GPT-2 is employed to generate $|\mathcal{O}^+|$ sentences where $\mathcal{O}^+$ is the set of sentences with at least one event mention in $\mathcal{O}$ (i.e., positive samples). We denote the generated dataset by $\mathcal{G}$. Note that our generation process ensures all the sentences in $\mathcal{G}$ to involve at least one marked event trigger from GPT-2.

To evaluate the quality of the generated data, we manually assess 100 sentences that are generated by GPT-2 after being fine-tuned on the ODED dataset LitBank [33]. Among these samples, we find that 85% of the sentences are grammatically correct, semantically sound, and similar to the original sentences in LitBank. Also, 80% of the marked event triggers in the sentences are correct. Refer to Section 3.5 for examples of generated sentences, noises and more data analysis.

2.2 Task Modeling

Our generated data in $\mathcal{G}$ is noisy as the sentences might not be natural and the event triggers might be incorrectly marked. To effectively combine $\mathcal{G}$ with the original data $\mathcal{O}$, this section describes our proposed teacher-student framework to overcome the noise in the generated data to train ODED models.

Base Model: The student and teacher models in our framework employ the same base network architecture for ODED (with different parameters). As such, following the recent ED model in [36], our base architecture for ODED starts with the pre-trained BERT model [6] whose output representations are consumed by a Bi-directional LSTM (BiLSTM), and then followed by a feed-forward network to make event predictions for each word in S. In particular, we first feed the input sentence $S = [w_1, w_2, \dots, w_N]$ into the pre-trained BERT$_{base}$ model. Afterward, the hidden vectors from the last layer of the BERT model are consumed by BiLSTM (i.e., the encoder) to generate the word representations $H = [h_1, h_2, \dots, h_N]$. The representation vector h_i for $w_i \in S$ is then sent to a two-layer feed-forward network with the sigmoid function in the end to obtain a probability distribution $P(\cdot|S, i)$ over possible event labels for w_i, i.e., *EVENT* or *NONE*. Finally, we use the negative log-likelihood as the loss function to train our models for ODED: $\mathcal{L}_{pred} = -\sum_{i=1}^{N} \log P(y_i|S, i)$ where y_i is the golden event label for $w_i \in S$.

Knowledge Consistency Enforcement: As motivated in the introduction, the teacher model $\mathcal{T}$ in our framework is first trained on the original noiseless data $\mathcal{O}$ to capture the anchor knowledge (using the loss $\mathcal{L}_{pred}$). The data combination $\mathcal{O} \cup \mathcal{G}$ will then be utilized to train the student model $\mathcal{S}$, also serving as our final model for ODED. To mitigate the noise from $\mathcal{G}$, the student model $\mathcal{S}$ would be regularized to be consistent with the anchor knowledge from the teacher. As such, we employ two types of anchor knowledge for teacher-student consistency, i.e., the representations of the input sentences and the difference between the synthetic and original data as perceived by the teacher.

First, to enforce the teacher-student compatibility based on representations of input sentences, we first feed the input sentence S into both the teacher and the student networks, and compute the representations of the sentence by max-pooling the representation vectors returned by the BiLSTM encoders for the words, i.e., $h^{\mathcal{T}} = MAX\_POOL[h_1^{\mathcal{T}}, \ldots, h_N^{\mathcal{T}}]$ for the teacher and similarly $h^{\mathcal{S}}$ for the student. The goal of the representation compatibility is to encourage the similarity between $h^{\mathcal{T}}$ and $h^{\mathcal{S}}$ for the same S. As such, one simple method is to directly compute the distance between the two vectors and use this distance as an auxiliary training loss. However, this might restrict the learning capability of the student as less room for variation in $h^{\mathcal{S}}$ is granted. To allow more variation in $h^{\mathcal{S}}$ and still maintain essential similarity with $h^{\mathcal{T}}$, we propose to instead maximize the mutual information (MI) between $h^{\mathcal{T}}$ and $h^{\mathcal{S}}$. However, computing MI between $h^{\mathcal{T}}$ and $h^{\mathcal{S}}$ is intractable due their high dimensionality. To address this issue, following [8], we treat $h^{\mathcal{T}}$ and $h^{\mathcal{S}}$ as random variables and estimate their MI via the Jenson-Shanon divergence between the joint distribution, i.e., $P_{T,S}(h^{\mathcal{T}}, h^{\mathcal{S}})$, and the product of the marginal distributions, i.e., $P_T(h^{\mathcal{T}}) * P_S(h^{\mathcal{S}})$. In particular, the Jenson-Shanon divergence computation can be fulfilled by a feed-forward discriminator, denoted by D, which distinguishes the samples of the joint distribution $P_{T,S}(h^{\mathcal{T}}, h^{\mathcal{S}})$ from those of the product distribution $P_T(h^{\mathcal{T}}) * P_S(h^{\mathcal{S}})$. In this work, we sample from $P_{T,S}(h^{\mathcal{T}}, h^{\mathcal{S}})$ by directly concatenating $h^{\mathcal{T}}$ and $h^{\mathcal{S}}$. The samples from $P_T(h^{\mathcal{T}}) * P_S(h^{\mathcal{S}})$ are obtained by concatenating $h^{\mathcal{T}}$ with the student-based representation vector for another sentence randomly chosen from the same mini-batch, i.e., $h'^{\mathcal{S}}$. Finally, the samples are sent to the discriminator D to produce scores for whether the samples are from the joint distribution or not. The negative log-likelihoods of the samples are then included in the overall loss function to serve as a way to maximize the MI between $h^{\mathcal{S}}$ and $h^{\mathcal{T}}$:

$$\mathcal{L}_{MI} = -\log(D([h^{\mathcal{T}}, h^{\mathcal{S}}])) - \log(1 - D([h^{\mathcal{T}}, h'^{\mathcal{S}}])) \tag{1}$$

Our second teacher-student consistency for learning is realized by the difference between synthetic and original data in $\mathcal{G}$ and $\mathcal{O}$. As such, our principle is to treat the teacher-based distance between the sentences in $\mathcal{G}$ and $\mathcal{O}$ as an anchor. The student should then adhere to this anchor distance to avoid significant divergence from the teacher for noise mitigation. Given that, the major question is how to effectively estimate the distance/difference between $\mathcal{G}$ and $\mathcal{O}$ for this consistency regularization. To this end, as presented in the introduction, our intuition is to simultaneously consider the representations and the event-containing likelihoods of the sentences in $\mathcal{G}$ and $\mathcal{O}$ for this distance function. To implement this intuition, we first discuss how we compute the distance $D^{\mathcal{S}}_{\mathcal{G},\mathcal{O}}$ between $\mathcal{G}$ and $\mathcal{O}$ for the student model $\mathcal{S}$.

In particular, given the student network $\mathcal{S}$, we first obtain a representation vector $h^{\mathcal{S}}$ for each sentence S in $\mathcal{G} \cup \mathcal{O}$ as done in the MI estimation above.

Afterward, to obtain an event-containing likelihood score $p^{\mathcal{S}}$ for S, we compute the maximum of the probabilities for being an event trigger of the words $w_i \in S$: $p^{\mathcal{S}} = \max_{i=1..N}(P^{\mathcal{S}}(EVENT|S,i))$ where $P^{\mathcal{S}}(\cdot|S,i)$ is the probability distribution over the possible event labels for w_i based on the student model $\mathcal{S}$. To facilitate the distance estimation for $\mathcal{G}$ and $\mathcal{O}$, let $R^o = [h_1^{\mathcal{S},o}, \ldots, h_n^{\mathcal{S},o}]$ and $P^o = [p_1^{\mathcal{S},o}, \ldots, p_n^{\mathcal{S},o}]$ be the sets of representation vectors and event-containing likelihood scores (respectively) for the sentences in $\mathcal{O}$ ($n = |\mathcal{O}|$). Similarly, let $R^g = [h_1^{\mathcal{S},g}, \ldots, h_m^{\mathcal{S},g}]$ and $P^g = [p_1^{\mathcal{S},g}, \ldots, p_m^{\mathcal{S},g}]$ be the sets of representation vectors and event-containing likelihood scores (respectively) for the sentences in $\mathcal{G}$ ($m = |\mathcal{G}|$). Consequently, to simultaneously exploit R^o, P^o, R^g and P^g for distance estimation, we seek to find an optimal alignment between sentences in $\mathcal{G}$ and $\mathcal{O}$ such that two sentences with closest representations and event-containing likelihoods have better chance to be aligned to each other. This problem can be then solved naturally with optimal transport (OT) methods that enable the computation of the optimal mapping between two probability distributions.

Formally, given the probability distributions $p(x)$ and $q(y)$ over the domains $\mathcal{X}$ and $\mathcal{Y}$, and the cost function $C(x,y) : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}_+$ for mapping $\mathcal{X}$ to $\mathcal{Y}$, OT finds the optimal joint distribution $\pi^*(x,y)$ (over $\mathcal{X} \times \mathcal{Y}$) with marginals $p(x)$ and $q(y)$, i.e., the cheapest transportation of $p(x)$ to $q(y)$, by solving the following problem:

$$\begin{gathered} \pi^*(x,y) = \min_{\pi \in \Pi(x,y)} \int_{\mathcal{Y}} \int_{\mathcal{X}} \pi(x,y) C(x,y) dx dy \\ \textbf{s.t. } x \sim p(x) \text{ and } y \sim q(y), \end{gathered} \tag{2}$$

where $\Pi(x,y)$ is the set of all joint distributions with marginals $p(x)$ and $q(y)$. Note that if the distributions $p(x)$ and $q(y)$ are discrete, the integrals in Equation 2 are replaced with a sum and the joint distribution $\pi^*(x,y)$ is represented by a matrix whose entry $\pi(x_i, y_j)$ represents the probability of transforming the data point x_i to y_j to convert the distribution $p(x)$ to $q(y)$. To this end, our model defines the domains $\mathcal{X}$ and $\mathcal{Y}$ for OT via the representation spaces of the sets R^g and R^o. As such, the cost function $C(x,y)$ is defined by the Euclidean distance between the representation vectors of the corresponding elements, i.e., $C(h_i^{\mathcal{S},g}, h_j^{\mathcal{S},o}) = \|h_i^{\mathcal{S},g} - h_j^{\mathcal{S},o}\|$. Also, the probability distributions $p(x)$ and $q(y)$ are defined over the normalized likelihood scores P^g and P^o, i.e., $p(h_i^{\mathcal{S},g}) = softmax(P^g)$ and $p(h_i^{\mathcal{S},o}) = softmax(P^o)$. Based on these definitions, the optimal transport $\pi^*(h_i^{\mathcal{S},g}, h_j^{\mathcal{S},o})$, which is obtained by solving Equation 2, could be used to compute the smallest transportation cost between $\mathcal{G}$ and $\mathcal{O}$ (i.e., Wasserstein distance), serving as the distance function $D^{\mathcal{S}}_{\mathcal{G},\mathcal{O}}$ in our model: $D^{\mathcal{S}}_{\mathcal{G},\mathcal{O}} = \Sigma_{i=1}^m \Sigma_{j=1}^n \pi^*(h_i^{\mathcal{S},g}, h_j^{\mathcal{S},o}) \|h_i^{\mathcal{S},g} - h_j^{\mathcal{S},o}\|$.

Note that as solving the OT problem in Equation 2 is intractable, we employ the entropy-based approximation of OT and solve it with the Sinkhorn algorithm [30].

In the next step, we apply the same procedure obtain the distance $D^{\mathcal{T}}_{\mathcal{G},\mathcal{O}}$ between $\mathcal{G}$ for the teacher network. Finally, to realize the $\mathcal{G}$-$\mathcal{O}$ distance consistency between the student and the teacher, we introduce the difference $\mathcal{L}_{diff}$ between

$D^{\mathcal{T}}_{\mathcal{G},\mathcal{O}}$ and $D^{\mathcal{S}}_{\mathcal{G},\mathcal{O}}$ into the overall loss function:

$$\mathcal{L}_{diff} = |D^{\mathcal{T}}_{\mathcal{G},\mathcal{O}} - D^{\mathcal{S}}_{\mathcal{G},\mathcal{O}}| \tag{3}$$

Note that in the actual implementation, we only compute $\mathcal{L}_{diff}$ for each mini-batch and add it to the loss function to enable efficient solving of the OT optimization.

Dynamic Sample Weighting: To further improve the representation learning of the student model for ODED, we seek to control the contribution of each generated sample in $\mathcal{G}$ for the overall training loss. In particular, we weight each generated sample in the loss function based on an estimated degree to which the sample can provide new training signals for the original data $\mathcal{O}$. In our model, the weight for each synthetic sample is determined based on its diversity score w.r.t. to other generated sentences in $\mathcal{G}$ and its novelty score w.r.t the original data. As such, we dynamically computes the weights during the training process where the weight and scores for each generated sample are updated immediately after each epoch to better capture the progress of the model.

Accordingly, to compute diversity scores for synthetic samples, we first cluster all generated sentences in $\mathcal{G}$ using their student-based representation vectors $h^{\mathcal{S},g}_i$ from the BiLSTM encoder and K-mean clustering[4]. This produces K centroid vectors $c^g = [c^g_1, \ldots, c^g_K]$ for K clusters $C^g = [C^g_1, \ldots, C^g_K]$ (respectively) in $\mathcal{G}$. Let $k(i)$ be the cluster index of the i-the sentence $S_i \in \mathcal{G}$ ($1 \leq k(i) \leq K$). The diversity score s^{div}_i for S_i in our model is then computed based on the distance between its representation $h^{\mathcal{S},g}_i$ and the corresponding centroid vector $c^g_{k(i)}$ (i.e., the farther S_i is toward the center, the more it contributes to the diversity in $\mathcal{G}$): $a_i = \|h^{\mathcal{S},g}_i - c^g_{k(i)}\|, s^{div}_i = e^{a_i} / \sum_{S_j \in C^g_{k(i)}} e^{a_j}$.

In the next step, we aim to compute novelty scores for the generated samples in $\mathcal{G}$ via their distances toward the original data $\mathcal{O}$ (i.e., the farther S_i is toward $\mathcal{O}$, the more novel it is for the training data). As such, we reuse the idea of OT-based distance between $\mathcal{G}$ and $\mathcal{O}$ to estimate the novelty score for the synthetic samples in $\mathcal{G}$ (i.e., based on both the representation vectors and the event-containing likelihoods of sentences). In particular, to enable efficient OT computation, we also first cluster the sentences in the original training dataset $\mathcal{O}$ using their representation vectors $h^{\mathcal{S},o}_i \in R^o$ from BiLSTM and K-mean clustering. The outputs from this clustering involve K centroid vectors $c^o = [c^o_1, \ldots, c^o_K]$ for the K clusters $C^o = [C^o_1, \ldots, C^o_K]$ (respectively) in $\mathcal{O}$. Afterward, we attempt to compute the optimal transport between the cluster sets C^g and C^o for the synthetic and original data whose results are leveraged to obtain novelty scores for generated samples later.

To this end, we directly use the centroid vectors in c^g and c^o as the representations for the clusters in C^g and C^o, serving as the domains $\mathcal{X}$ and $\mathcal{Y}$ for OT. Also, the event-containing likelihood score p^t_i for the cluster C^t_i ($t \in \{g, o\}$ to indicate synthetic or original data, $1 \leq i \leq K$) is estimated via the average of the student-based likelihood scores $p^{\mathcal{S},t}_j$ of the sentences in C^t_i: $p^t_i = \frac{1}{|C^t_i|} \sum_{S_j \in C^t_i} p^{\mathcal{S},t}_j$.

[4] $K = 10$ produces the best performance in our study.

Given the representations and event-containing likelihood scores, we follow the same described procedure to obtain the optimal transport $\pi^*_c(c^g_i, c^o_j)$ between the cluster sets C^g and C^o (i.e., by solving Equation 2 and using the Euclidean distance for the cost function $C(x, y)$). In the next step, we align each cluster C^g_i in $\mathcal{G}$ to cluster $C^o_{i^*}$ in $\mathcal{O}$ that has the highest probability in the optimal transport, i.e., $i^* = \text{argmax}_j\, \pi^*_c(c^g_i, c^o_j)$. The distance d_i between the cluster C^g_i in $\mathcal{G}$ toward the original data $\mathcal{O}$ is then obtained via the distance between the centroid vectors of C^g_i and $C^o_{i^*}$: $b_i = \|c^g_i - c^o_{i^*}\|, d_i = e^{b_i} / \sum_{j=1..K} e^{b_j}$.

As such, the novelty score s^{nov}_i for a generated sample $S_i \in \mathcal{G}$ in our work will be set directly to the distance of its corresponding cluster $C^g_{k(i)}$ toward the original data: $s^{nov}_i = d_{k(i)}$.

Finally, to aggregate the diversity and novelty scores for each sentence $S_i \in \mathcal{G}$ to compute its weight s^{comb}_i in the training loss, we set $s^{comb}_i = (1-\alpha)s^{nov}_i + \alpha s^{div}_i$ and normalize it over the generated samples (α is a trade-off parameter). For convenience, we consider the examples in the original data $\mathcal{O}$ as having the weight of 1. The final training loss function for the student network over an example $S \in \mathcal{G} \cup \mathcal{O}$ with the weight s^{comb} is thus: $\mathcal{L} = s^{comb} * \mathcal{L}_{pred} + \beta * \mathcal{L}_{MI} + \gamma * \mathcal{L}_{diff}$ where β and γ are the trade-off parameters.

3 Experiments

3.1 Datasets & Hyper-parameters

We use the following three ODED datasets to evaluate the models[5]:

- **LitBank** [33]: This dataset provides event annotation for 100 English literary texts. We use the same data split (train/dev/test) as [33] for fair comparison.
- **TimeBank** [31]: This dataset annotates 183 English news articles with event mentions and temporal relations. To make fair comparison with prior work [22], we exclude event mentions in TimeBank that have not occurred (e.g., future, hypothetical or negated events) and apply the the same data split.
- **SW100** [3]: This corpus presents event annotations for 100 documents in 10 different domains from Simple Wikipedia. The original data split in [3] is inherited in our experiment.

In our experiments we use the small version of GPT-2 to generate data. All the hyperparameters for the proposed model are selected based on the F1 scores on the development set of LitBank. The same hyper-parameters from this fine-tuning are then applied for the other datasets for consistency. In the base model, we use the BERT$_{base}$ version, 200 dimensions in the hidden states of the BiLSTM, and 2 layers for the feed-forward neural network with 200 hidden

[5] Note that we do not use the ACE 2005 dataset [35] as it only focuses on a small set of event types in the news domain, thus being not appropriate for our open-domain setting of event detection.

	LitBank	TimeBank	SW100
BiLSTM-Base$_{\mathcal{O}}$	73.4	81.9	85.3
BiLSTM-Base$_{\mathcal{G}}$	75.9	80.0	83.8
BiLSTM-Base$_{\mathcal{O}+\mathcal{G}}$	74.0	82.3	84.3
Confidence-Filtering [2]	73.9	82.5	86.1
Novelty-Filtering [39]	74.9	81.4	86.8
BiLSTM+BERT* [33]	73.9	-	-
GatedGCN* [15]	74.5	-	-
BERT-A* [22]	-	82.6	-
DS-BiLSTM* [3]	-	-	72.4
GPT-Augmented (proposed)	**77.4**	**85.0**	**89.7**

Table 1. Model performance (F1 scores) on the test data of the datasets. Rows with * indicate results taken from the corresponding papers.

dimensions to predict events. The discriminator D of the MI component consists of an one-layer feed-forward neural network with 200 hidden dimensions. The trade-off parameters α, β and γ are set to 0.6, 0.1, and 0.05, respectively. The learning rate is set to 0.3 for the Adam optimizer and the batch size of 50 is employed during training. Finally, note that we do not update the BERT model for word embeddings in this work due to its better performance on the development data of LitBank. We use a single GeForce RTX 2080 GPU with 11GB memory to run the models in this work. PyTorch 1.1 is used to implement the models.

3.2 Baselines

We compare our model (called **GPT-Augmented**) with the following baselines:

• Base models: The baselines in this group all employ the base architecture in Section 2.2. Three versions of this base model are possible, i.e., **BiLSTM-Base**$_{\mathcal{O}}$, **BiLSTM-Base**$_{\mathcal{G}}$, and **BiLSTM-Base**$_{\mathcal{O}+\mathcal{G}}$, depending on whether it is trained on the original training data $\mathcal{O}$, the generated data $\mathcal{G}$, or the combination of $\mathcal{O}$ and $\mathcal{G}$ respectively.

• Noisy mitigation methods: We consider two recent methods that are proposed to mitigate the noise from the GPT-generated data to train models: (i) **Confidence-Filtering**: Inspired by LAMBADA [2], this baseline filters the generated data based on the confidence of the trained base model BiLSTM-Base$_{\mathcal{O}}$ in predicting labels for the generated data. Specifically, BiLSTM-Base$_{\mathcal{O}}$ is first employed to make predictions on the generated data. The synthetic sentences in $\mathcal{G}$ are then sorted via the confidence of the model (i.e., the event-containing likelihoods with the event probabilities from BiLSTM-Base$_{\mathcal{O}}$). Only sentences with a confidence above a tuned threshold are kept and combined with the original data to retrain the base model; and (ii) **Novelty-Filtering**: Inspired by [39], this baseline also filters the generated samples in $\mathcal{G}$ as Confidence-Filtering; however, instead of using the confidence, it employs novelty scores. As such, the novelty score for one generated sentence $S \in \mathcal{G}$ is obtained via the average of the Euclidean distances between the representations of S and each sentence in the original training data $\mathcal{O}$. Here, the representations of the examples are also computed by the trained BiLSTM-Base$_{\mathcal{O}}$ model.

• State-of-the-art models: We also compare GPT-Augmented with the existing systems that report the state-of-the-art performance on each dataset. Namely, on LitBank we compare with the **BiLSTM+BERT** model in [33] and the recent **GatedGCN** model in [15]. On TimeBank, we compare with the in-domain performance of the **BERT-A** model in [22] that takes an adversarial domain adaptation approach. Finally, on SW100 dataset, we compare with the **DS-BiLSTM** model in [3] that augments the training data with distant supervision data.

3.3 Results

The performance (i.e., F1 scores) of the models on three datasets is shown in Table 1. As can be seen, the proposed model significantly outperforms the baselines (with $p < 0.01$) over all the three datasets. Specifically, compared to BiLSTM-Base$_{\mathcal{O}}$, GPT-Augmented improves F1 scores by at least 3%[6]. We attribute this improvement to the new training signals from the GPT-generated data and the effectiveness of the proposed techniques for noise mitigation and generated sample weighting for ODED. Also, the necessity of the proposed knowledge consistency enforcement for noise mitigation and sample weighting can be better revealed by considering the baselines BiLSTM-Base$_{\mathcal{O}+\mathcal{G}}$, Confidence-Filtering, and Novelty-Filtering that all exploit the generated data $\mathcal{G}$, but considerably under-perform the proposed model.

3.4 Ablation Study

This section studies the contribution of different components of the proposed model. We analyze the performance of GPT-Augmented on the LitBank development set when its two major components are removed or altered, i.e., (1) Knowledge Consistency Enforcement (KCE) and (2) Dynamic Sample Weighting (DSW).

Knowledge Consistency Enforcement: First, for KCE, we examine the following baselines:

(i) Full-MI: This model excludes the representation consistency based on MI from GPT-Augmented;

(ii) Full-MI+Euclidean: This model replaces the MI-based loss $\mathcal{L}_{MI}$ with Euclidean loss in GPT-Augmented (i.e., $\|h^{\mathcal{S}} - h^{\mathcal{T}}\|$);

(iii) Full-OT: This model eliminates the OT-based loss $\mathcal{L}_{OT}$ for data difference compatibility from GPT-Augmented;

(iv) Full-OT^{rep}: This baseline still employs the OT-based loss $\mathcal{L}_{OT}$; however, instead of computing the cost function $C(h_i^{\mathcal{S},g}, h_j^{\mathcal{S},o})$ based on representation

[6] In the experiments, we learn that augmenting the models with GPT-generated data is more helpful for recalls.

Model	P	R	F1
GPT-Augmented (Full)	78.9	82.0	**80.4**
Full-MI	74.1	84.6	79.1
Full-MI+Euclidean	74.4	84.4	79.1
Full-OT	74.0	84.4	78.9
Full-OT^{rep}	75.9	82.6	79.1
Full-OT^{event}	76.1	79.8	77.9
Full-OT+Euclidean	76.1	81.8	78.9
Full-OT-MI	74.3	82.2	78.1

Table 2. Performance on the LitBank development set.

distances, this model uses a constant cost function, i.e., $C(h_i^{\mathcal{S},g}, h_j^{\mathcal{S},o}) = 1$. This baseline aims to show that sentence representations play an important role for distance estimation between synthetic and original data for ODED;

(v) Full-OT^{event}: Instead of using event-containing likelihoods of sentences to compute the distributions $p(h_i^{\mathcal{S},g})$ and $p(h_i^{\mathcal{S},o})$ in the OT-based loss for GPT-Augmented, this method utilizes uniform distributions. The goal of this baseline is to demonstrate that event-containing likelihoods of sentences are crucial for the distance estimation between synthetic and original data for ODED;

(vi) Full-OT+Euclidean: This baseline completely replaces the OT loss $\mathcal{L}_{diff}$ with the Euclidean distance between the original data $\mathcal{O}$ and the generated data $\mathcal{G}$ for data difference consistency in KCE, i.e., the distance $D^{\mathcal{S}}_{\mathcal{G},\mathcal{O}}$ for the student network $\mathcal{S}$ is computed by: $D^{\mathcal{S}}_{\mathcal{G},\mathcal{O}} = \frac{1}{mn}\Sigma_{i=1}^{m}\Sigma_{j=1}^{n}\|h_i^{\mathcal{S},g} - h_j^{\mathcal{S},o}\|$;

(vii) Full-OT-MI: This baseline completely removes the OT- and MT-based loss; it excludes the KCE component and the teacher $\mathcal{T}$.

Table 2 shows the performance of the baseline models for KCE. It is clear from the table that both MI- and OT-based losses for KCE are important for GPT-Augmented to achieve its best performance. Comparing GPT-Augmented and Full-MI+Euclidean, we see that using MI to realize the representation-based KCE is significantly better than its alternatives (i.e., via Euclidean distance) for GPT-Augmented. Finally, comparing different methods to estimate the distance between the generated and the original data for KCE (i.e., Full-OT^{rep}, Full-OT^{event}, and Full-OT+Euclidean), the OT-based approach in GPT-Augmented clearly demonstrates its benefits with the highest performance. This testifies to the importance of both sentence representations and event-containing likelihoods for the computation of the distance between $\mathcal{G}$ and $\mathcal{O}$ for ODED.

Dynamic Sample Weighting: Second, for the DSW component, we study the contribution of the proposed techniques for the sample weights with diversity and novelty scores. To this end, we evaluate the following models:

(i) Full-Div: This baseline excludes the diversity score from the weight of each generated sentence in the training loss of GPT-Augment, i.e., $s_i^{comb} = s_i^{nov}$.;

(ii) Full-Nov: This baseline excludes the novelty score s_i^{nov} from GPT-Augmented, i.e., $s_i^{comb} = s_i^{div}$.;

Model	P	R	F1
GPT-Augmented (Full)	78.9	82.0	**80.4**
Full-Div	78.1	80.4	79.2
Full-Nov	77.9	80.1	79.0
Full-Nov^{rep}	76.9	79.4	78.1
Full-Nov^{event}	77.2	80.6	78.8
Full-Cluster	74.2	82.6	78.2
Full-Div-Nov	75.1	82.9	78.8

Table 3. Performance on the LitBank development set.

(iii) Full-Nov^{rep}: In the computation of novelty scores using OT, this baseline replaces the cost function based on the centroid vectors with a constant function, i.e., $C(c_i^g, c_j^o) = 1$.;

(iv) Full-Nov^{event}: This model employs uniform distributions for the probability distributions in the OT-based computation for novelty scores (i.e., ignoring the event-containing likelihoods).;

(v) Full-Cluster: This baseline aims to completely eliminate the data clustering in GPT-Augmented.;

(vi) Full-Div-Nov: This component entirely removes DSW, using the same weight for all the samples, i.e., $s_i^{comb} = 1$.

The performance of the baseline models for DSW are presented in Table 3. The Table shows that both novelty and diversity scores are important for the best performance of GPT-Augmented. In addition, computing these scores without clustering cannot reach the best performance, indicating the necessity of clustering the data before computing the scores. Finally, the superiority of GPT-Augmented over Full-Nov^{rep} and Full-Nov^{event}, emphasizes the benefits of leveraging both representations and event-containing likelihoods of data clusters with OT to obtain the distance between original and GPT-generated data for ODED.

3.5 Analysis

To further evaluate the novelty/similarity of the event mention patterns captured in the generated data $\mathcal{G}$ w.r.t the original data $\mathcal{O}$, we train the base model in Section 2.2 only on the original data $\mathcal{O}$ (i.e., BiLSTM-Base$_{\mathcal{O}}$) and directly evaluate its performance on the generated data $\mathcal{G}$. The precision, recall and F1 scores for this experiment using the LitBank dataset are 94.9%, 87.1% and 90.8%, respectively. As such, the high precision of BiLSTM-Base$_{\mathcal{O}}$ on $\mathcal{G}$ shows that the generated data is highly compatible with the original data. More importantly, the lower recall indicates that the generated data does involve event mention patterns that are novel for the original data. Therefore, training ODED models on the combination of generated and original data could improve the models' performance, as demonstrated in our experiments.

To understand the effect of the generated data size for GPT-Augmented, Table 4 reports the performance of GPT-Augmented on the development data

$\|\mathcal{G}\|$	P	R	F1
$0.5 * \|\mathcal{O}^+\|$	77.6	80.6	79.1
$1.0 * \|\mathcal{O}^+\|$	78.9	82.0	80.4
$2.0 * \|\mathcal{O}^+\|$	75.0	81.6	78.2
$3.0 * \|\mathcal{O}^+\|$	76.2	76.0	76.1

Table 4. The performance of GPT-Augmented on the LitBank dev set with different sizes of the generated data $\mathcal{G}$.

of LitBank when we vary the size of the generated data $\mathcal{G}$. On the one hand, we find that decreasing the size $|\mathcal{G}|$ of the generated data to half of positive samples $\mathcal{O}^+$ in the original data $\mathcal{O}$ cannot fully benefit from the generated data from GPT-2. On the other hand, increasing the size for $\mathcal{G}$ (i.e., to two or three times larger than $\mathcal{O}^+$) could introduce more noises and hurt the model performance.

Table 6 presents some noiseless examples generated from GPT-2 while Table 5 shows some noisy sentences along with their categories in the generated data.

Error	Sentence
Inconsistency	I TRG$_s$ **heard** TRG$_e$ that he was very fond of women and I had a good time at the dinner table; but in the mean time she TRG$_s$ **told** TRG$_e$ me in the end it was all about her!
Repetition	“That’s my boy” TRG$_s$ **said** TRG$_e$ the little boy ”and that’s my boy”.
Meaninglessness	He TRG$_s$ **looked** TRG$_e$ down at his watch and then at the clock in his shoes.
Missing Triggers	He TRG$_s$ **left** TRG$_e$ London yesterday, at 2 P.M, and arrived at Paris two hours later;
Incorrect Triggers	There has been no TRG$_s$ **revolution** TRG$_e$ in this country since 1960.

Table 5. Samples of noisy generated sentences for the LitBank dataset from GPT-2. Event triggers are shown in boldface and surrounded with special tokens TRG$_s$ and TRG$_e$, generated by GPT-2. In the first example, i.e., “*Inconsistency*” error, the model changes the pronoun from *he* to *she* in the first and second clause in the sentence. For the error of “*Missing Triggers*”, GPT-2 fails to mark the word *arrived* as an event trigger. Finally, in the example for the error of “*Incorrect Triggers*”, there is no event trigger in the sentence and GPT-2 incorrectly marks “*revolution*” as an event trigger.

Dataset	Sentence
LitBank	I was excited to go with him, as we had TRG$_s$ **met** TRG$_e$ earlier before this.
LitBank	We TRG$_s$ **went** TRG$_e$ out in groups and TRG$_s$ **came** TRG$_e$ after a long journey.
TimeBank	Last year, Beijing companies TRG$_s$ **proposed** TRG$_e$ more TRG$_s$ **discounts** TRG$_e$ than western companies.
TimeBank	Some experts TRG$_s$ **anticipate** TRG$_e$ that the long TRG$_s$ **recession** TRG$_e$ TRG$_s$ **resulted** TRG$_e$ in all absolute TRG$_s$ **loss** TRG$_e$ for all players.
SW100	Scientists have TRG$_s$ **found** TRG$_e$ that this plant is so small that no rat has been TRG$_s$ **fed** TRG$_e$ by it.
SW100	40 years ago, the British government TRG$_s$ **entered** TRG$_e$ into an TRG$_s$ **agreement** TRG$_e$ with all groups in TRG$_s$ **rebellions** TRG$_e$.

Table 6. Samples of generated sentences for each dataset. Event triggers are shown in boldface and surrounded with special tokens TRG$_s$ and TRG$_e$, generated by GPT-2.

4 Related Work

The major approaches for ED involve early feature-based models [1, 11, 17, 18, 20, 16, 21, 37] and recent deep learning models [5, 26, 43, 38, 27, 44, 14, 34, ?]. Open domain event detection has attracted more attentions recently [3, 33, 22].

A challenge for ED in general and for ODED in particular is the scarcity of large training datasets. As such, prior work has attempted to address this issue via unsupervised [9, 40], semi-supervised [17, 10, 7], distantly supervised [23, 41, 3] or domain adaptation [22] models. In this work, we take a novel approach by utilizing the pre-trained language model GPT-2 to generate new training data for ODED.

Using GPT-2 to generate training data has been studied very recently for other NLP tasks, including relation extraction [28], multi-label text classification [42], commonsense reasoning [39], event influence prediction [19], knowledge base completion [4], sentiment, intent and question classification [13, 2], and spoken language understanding [29]. In order to deal with noisy generated data, these approaches have only heuristically and statically filtered out noisy data. Unlike such prior work, our model keeps all the generated data and introduce a novel teacher-student framework to allow models learn with noisy data, featuring MI- and OT-based KCE and DSW mechanisms.

5 Conclusion

We propose a novel approach to generate training data for ODED by fine-tuning the pre-trained language model GPT-2. To deal with the noises in the generated data, we propose a teacher-student framework in which the teacher model is used to capture anchor knowledge (i.e., sentence representations and synthetic-original data difference) to regularize the student model. We present novel mechanisms to encourage the knowledge consistency between the student and the teacher based on MI and OT. We also introduce a dynamic method to weight the generated sentences. In the future, we plan to apply the proposed method to other related NLP tasks.

Acknowledgments

This research has been supported by the Army Research Office (ARO) grant W911NF-21-1-0112 and the NSF grant CNS-1747798 to the IUCRC Center for Big Learning. This research is also based upon work supported by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via IARPA Contract No. 2019-19051600006 under the Better Extraction from Text Towards Enhanced Retrieval (BETTER) Program. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of ARO, ODNI, IARPA, the Department of Defense, or the U.S. Government.

References

1. Ahn, D.: The stages of event extraction. In: Proceedings of the Workshop on Annotating and Reasoning about Time and Events (2006)
2. Anaby-Tavor, A., Carmeli, B., Goldbraich, E., Kantor, A., Kour, G., Shlomov, S., Tepper, N., Zwerdling, N.: Do not have enough data? deep learning to the rescue! In: AAAI (2020)
3. Araki, J., Mitamura, T.: Open-domain event detection using distant supervision. In: COLING (2018)
4. Bosselut, A., Rashkin, H., Sap, M., Malaviya, C., Celikyilmaz, A., Choi, Y.: Comet: Commonsense transformers for automatic knowledge graph construction. In: ACL (2019)
5. Chen, Y., Xu, L., Liu, K., Zeng, D., Zhao, J.: Event extraction via dynamic multi-pooling convolutional neural networks. In: ACL-IJCNLP (2015)
6. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: Pre-training of deep bidirectional transformers for language understanding. In: NAACL-HLT (2019)
7. Ferguson, J., Lockard, C., Weld, D.S., Hajishirzi, H.: Semi-supervised event extraction with paraphrase clusters. In: NAACL (2018)
8. Hjelm, R.D., Fedorov, A., Lavoie-Marchildon, S., Grewal, K., Bachman, P., Trischler, A., Bengio, Y.: Learning deep representations by mutual information estimation and maximization. In: ICLR (2019)
9. Huang, L., Cassidy, T., Feng, X., Ji, H., Voss, C., Han, J., Sil, A.: Liberal event extraction and event schema induction. In: ACL (2016)
10. Huang, R., Riloff, E.: Bootstrapped training of event extraction classifiers. In: EACL (2012)
11. Ji, H., Grishman, R.: Refining event extraction through cross-document inference. In: ACL (2008)
12. Keith, K., Handler, A., Pinkham, M., Magliozzi, C., McDuffie, J., O'Connor, B.: Identifying civilians killed by police with distantly supervised entity-event extraction. In: EMNLP (2017)
13. Kumar, V., Choudhary, A., Cho, E.: Data augmentation using pre-trained transformer models. arXiv preprint arXiv:2003.02245 (2020)
14. Lai, V.D., Dernoncourt, F., Nguyen, T.H.: Exploiting the matching information in the support set for few shot event classification. In: Proceedings of the 24th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD) (2020)
15. Lai, V.D., Nguyen, T.N., Nguyen, T.H.: Event detection: Gate diversity and syntactic importance scores for graph convolution neural networks. In: EMNLP (2020)
16. Li, Q., Ji, H., Huang, L.: Joint event extraction via structured prediction with global features. In: ACL (2013)
17. Liao, S., Grishman, R.: Filtered ranking for bootstrapping in event extraction. In: COLING (2010)
18. Liao, S., Grishman, R.: Using document level cross-event inference to improve event extraction. In: ACL (2010)
19. Madaan, A., Rajagopal, D., Yang, Y., Ravichander, A., Hovy, E., Prabhumoye, S.: Eigen: Event influence generation using pre-trained language models. arXiv preprint arXiv:2010.11764 (2020)
20. McClosky, D., Surdeanu, M., Manning, C.: Event extraction as dependency parsing. In: BioNLP Shared Task Workshop (2011)
21. Miwa, M., Thompson, P., Korkontzelos, I., Ananiadou, S.: Comparable study of event extraction in newswire and biomedical domains. In: COLING (2014)

22. Naik, A., Rosé, C.: Towards open domain event trigger identification using adversarial domain adaptation. In: ACL (2020)
23. Nguyen, M., Nguyen, T.H.: Who is killed by police: Introducing supervised attention for hierarchical lstms. In: COLING (2018)
24. Nguyen, T.H., Cho, K., Grishman, R.: Joint event extraction via recurrent neural networks. In: NAACL (2016a)
25. Nguyen, T.H., Grishman, R.: Event detection and domain adaptation with convolutional neural networks. In: ACL (2015b)
26. Nguyen, T.H., Grishman, R.: Graph convolutional networks with argument-aware pooling for event detection. In: AAAI (2018)
27. Nguyen, T.M., Nguyen, T.H.: One for all: Neural joint modeling of entities and events. In: AAAI (2019)
28. Papanikolaou, Y., Pierleoni, A.: Dare: Data augmented relation extraction with gpt-2. In: SciNLP workshop at AKBC (2020)
29. Peng, B., Zhu, C., Zeng, M., Gao, J.: Data augmentation for spoken language understanding via pretrained models. arXiv preprint arXiv:2004.13952 (2020)
30. Peyre, G., Cuturi, M.: Computational optimal transport: With applications to data science. In: Foundations and Trends in Machine Learning (2019)
31. Pustejovsky, J., Hanks, P., Sauri, R., See, A., Gaizauskas, R., Setzer, A., Radev, D., Sundheim, B., Day, D., Ferro, L., et al.: The timebank corpus. In: Corpus linguistics (2003)
32. Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I.: Language models are unsupervised multitask learners. OpenAI blog **1**(8), 9 (2019)
33. Sims, M., Park, J.H., Bamman, D.: Literary event detection. In: ACL (2019)
34. Trong, H.M.D., Le, D.T., Veyseh, A.P.B., Nguyen, T., Nguyen, T.H.: Introducing a new dataset for event detection in cybersecurity texts. In: EMNLP (2020)
35. Walker, C., Strassel, S., Medero, J., Maeda, K.: Ace 2005 multilingual training corpus. In: LDC, Philadelphia. (2006)
36. Wang, X., Han, X., Liu, Z., Sun, M., Li, P.: Adversarial training for weakly supervised event detection. In: NAACL-HLT (2019)
37. Yang, B., Mitchell, T.M.: Joint extraction of events and entities within a document context. In: NAACL-HLT (2016)
38. Yang, S., Feng, D., Qiao, L., Kan, Z., Li, D.: Exploring pre-trained language models for event extraction and generation. In: ACL (2019)
39. Yang, Y., Malaviya, C., Fernandez, J., Swayamdipta, S., Le Bras, R., Wang, J.P., Bhagavatula, C., Choi, Y., Downey, D.: Generative data augmentation for commonsense reasoning. In: Findings of EMNLP 2020 (2020)
40. Yuan, Q., Ren, X., He, W., Zhang, C., Geng, X., Huang, L., Ji, H., Lin, C.Y., Han, J.: Open-schema event profiling for massive news corpora. In: CIKM (2018)
41. Zeng, Y., Feng, Y., Ma, R., Wang, Z., Yan, R., Shi, C., Zhao, D.: Scale up event extraction learning via automatic training data generation. In: AAAI (2017)
42. Zhang, D., Li, T., Zhang, H., Yin, B.: On data augmentation for extreme multi-label classification. arXiv preprint arXiv:2009.10778 (2020)
43. Zhang, J., Qin, Y., Zhang, Y., Liu, M., Ji, D.: Extracting entities and events as a single task using a transition-based neural model. In: IJCAI (2019)
44. Zhang, Y., Xu, G., Wang, Y., Lin, D., Li, F., Wu, C., Zhang, J., Huang, T.: A question answering-based framework for one-step event argument extraction. In: IEEE Access, vol 8, 65420-65431 (2020)