Event Extraction as Multi-turn Question Answering

Fayuan Li, Weihua Peng, Yuguang Chen, Quan Wang, Lu Pan, Yajuan Lyu, Yong Zhu
Baidu Inc., China
{lifayuan, pengweihua, chenyuguang, wangquan05,
panlu01, lvyajuan, zhuyong }@baidu.com

Abstract

Event extraction, which aims to identify event triggers of pre-defined event types and their arguments of specific roles, is a challenging task in NLP. Most traditional approaches formulate this task as classification problems, with event types or argument roles taken as golden labels. Such approaches fail to model rich interactions among event types and arguments of different roles, and cannot generalize to new types or roles. This work proposes a new paradigm that formulates event extraction as multi-turn question answering. Our approach, MQAEE, casts the extraction task into a series of reading comprehension problems, by which it extracts triggers and arguments successively from a given sentence. A history answer embedding strategy is further adopted to model question answering history in the multi-turn process. By this new formulation, MQAEE makes full use of dependency among arguments and event types, and generalizes well to new types with new argument roles. Empirical results on ACE 2005 shows that MQAEE outperforms current state-of-the-art, pushing the final F1 of argument extraction to 53.4% (+2.0%). And it also has a good generalization ability, achieving competitive performance on 13 new event types even if trained only with a few samples of them.

1 Introduction

Event extraction is an important yet challenging task in natural language understanding. Given a sentence, an event extraction system ought to identify event triggers with specific event types, as well as their corresponding arguments with specific argument roles. As an example, Figure 1 presents an event mention of the type *Movement Transport*, triggered by "left". There are three arguments: "Saddam's family" playing the role of *Artifact*, "that city" the role of *Origin*, and "three days ago" the role of *Time-Within*.

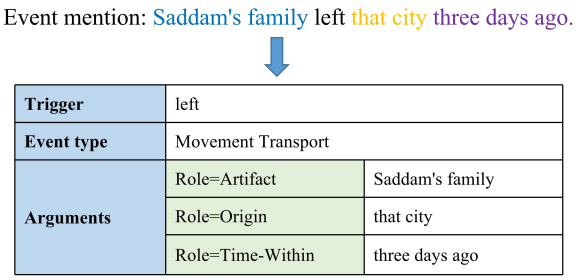

Figure 1: An example of event extraction.

Typically, event extraction can be divided into two subtasks: trigger extraction (trigger identification and classification) and argument extraction (argument identification and classification), as defined by the standard Automatic Content Extraction (ACE) 2005 benchmark (Grishman et al., 2005). Current approaches to event extraction can thus be roughly categorized into two groups: (1) pipelined approaches that perform trigger extraction and argument extraction in separate stages (Liao and Grishman, 2010; Hong et al., 2011; Lu and Roth, 2012; Chen et al., 2015; Yang et al., 2019); (2) joint approaches that perform all subtasks simultaneously in a joint learning fashion (Li et al., 2013; Nguyen et al., 2016; Liu et al., 2018; Sha et al., 2018).

Most of these approaches, whether pipelined or joint, formulate event extraction as classification tasks, by classifying event triggers into pre-defined event types (trigger extraction), and further event arguments into pre-defined argument roles (argument extraction). By treating event types and argument roles directly as golden labels, such classification-based approaches suffer from two limitations. First of all, they cannot explicitly model the semantics of these golden labels and also fail to capture the rich interactions among them, which could be extremely useful for event extraction. Consider the example in Figure 1. The event type *Movement Transport* actually provides valuable supplements

Findings of the Association for Computational Linguistics: EMNLP 2020, pages 829–838

to the corresponding argument roles like *Origin* and *Time-Within*. Besides, given that "Saddam's family" (the subject of the sentence) is an argument of the role *Artifact*, it is more likely to infer "that city" (the object) might be an argument of the role *Origin*. Effectively modeling these semantics and interactions would definitely be beneficial.

The second limitation lies in the generalization ability. By taking event types and argument roles as golden labels, classification-based approaches are not able to be generalized to new event types or argument roles without additional annotations. Huang et al. (2018) recently proposed a transfer learning architecture for zero-shot event extraction. The key idea of their approach is to represent event mentions and event types (or arguments and argument roles) in a shared semantic space, and cast trigger (or argument) classification as a semantic matching problem. This approach generalizes better to new event types and argument roles. But it can hardly capture full mention-type (or argument-role) interactions simply with the final cosine similarity matching. And it also relies heavily on structured features such as trees or paths derived by the AMR parser (Wang et al., 2015), prone to error propagation.

To address the above limitations, we propose a new paradigm that formulates event extraction as *multi-turn question answering* (QA). Our approach, referred to as MQAEE, splits event extraction into three sub-tasks: trigger identification, trigger classification, and argument extraction. These subtasks are modeled by a series of machine reading comprehension (MRC) based QA templates. Trigger identification is cast into an extractive MRC problem, identifying trigger words from given sentences. Trigger classification is formalized as a YES/NO QA problem, judging whether or not a candidate trigger belongs to a specific event type. Argument extraction is also solved via extractive MRC, with questions constructed iteratively by a target event type and the corresponding argument roles. Table 1 provides an example and overview of our approach. MQAEE has two major advantages: (1) The multi-turn QA infrastructure provides an effective way to model rich interactions among triggers, event types, and arguments, which has shown to be beneficial to event extraction. (2) By converting event types and argument roles as questions rather than golden labels, MQAEE can be easily generalized to new types and roles.

Passage: Saddam's family left that city three days ago.
Trigger identification
Q_1: Which word is the trigger word?
A_1: left
Trigger classification
Q_2: The trigger word is left $\langle pos \rangle$ 2 $\langle /pos \rangle$, movement: transport?
A_2: YES
Argument extraction
Q_3: left $\langle pos \rangle$ 2 $\langle /pos \rangle$. Movement:transport, time-within?
A_3: three days ago
Q_4: left $\langle pos \rangle$ 2 $\langle /pos \rangle$. Movement:transport, artifact?
A_4: Saddam's family
Q_5: left $\langle pos \rangle$ 2 $\langle /pos \rangle$. Movement:transport, destination?
A_5: NULL
. . .

Table 1: Example and overview of our MQAEE framework. Here the sentence is taken as the passage. Each turn contains a question (Q_i) and an answer (A_i). NULL means there is no answer to the question.

We evaluate our approach on the standard ACE 2005 benchmark. Experimental results show that MQAEE significantly outperforms current state-of-the-art, pushing the final F1 score of argument extraction to 53.4% (+2.0%). Moreover, MQAEE generalizes well to new event types, achieving competitive results on the 13 new event types even if trained only with the a few samples of them.

Our contributions are summarized as follows: (1) We propose a novel multi-turn QA framework for event extraction, which makes full use of rich interactions among triggers, event types, and arguments, and generalizes well to new event types. (2) We particularly apply the multi-turn QA idea to argument extraction, so as to capture the strong dependency among arguments of different roles associated with a same event type. (3) Empirical evaluation demonstrates the effectiveness and generalization ability of our approach.

2 Preliminaries

This section formally defines the event extraction task, and then introduces MRC techniques based on pre-trained language models (Devlin et al., 2018).

2.1 Task Definition

We follow the standard definition of event extraction, adopted by the ACE 2005 benchmark (Grishman et al., 2005). Presented below are the main terminologies.

- *Event mention*: a phrase or sentence within which an event is described, e.g., the sentence (denoted as S) given in Figure 1.
- *Event trigger*: the main word that most clearly expresses the occurrence of an event, e.g., the word "left" in S.
- *Event type*: the semantic class of an event, e.g., the event type of the trigger "left" in S is *Movement Transport*.
- *Event argument*: an entity mention, temporal expression or value that is involved in an event, e.g., "Saddam's family" in S.
- *Argument role*: the relation of an argument to the event in which it participates, e.g., the role of "Saddam's family" is *Artifact*.

Given an sentence, an extraction system have to identify event triggers and assign event types to the identified triggers (trigger extraction), and then identify event arguments and assign the roles to them (argument extraction). Golden entities, including entity mentions, temporal expressions and values are provided to the extraction system in the ACE 2005 benchmark. Most previous approaches adopt golden entities as candidate arguments. As in realistic scenario golden entities are not available, our approach extracts triggers and arguments without considering golden entities.

2.2 MRC with BERT

MRC is an important NLP task where the machine is required to answer questions about a given passage. This paper considers two types of MRC: (1) *extractive-style* where the answers are constrained as contiguous spans in the passage; (2) *YES/NO-style* where the answers are restricted to "yes" or "no". Recent years have seen remarkable success in the application of pre-trained language models, e.g., BERT (Devlin et al., 2018), to MRC, which achieves new state-of-the-art performance across various benchmarks. Next, we formally describe MRC based on BERT, a core module of the whole MQAEE framework.

Suppose we are given a passage P with m tokens and a question Q with n tokens. The question and the passage are packed into a single sequence $C = [\langle \texttt{CLS} \rangle, Q, \langle \texttt{SEP} \rangle, P, \langle \texttt{SEP} \rangle]$, where $\langle \texttt{SEP} \rangle$ is the separating token, and $\langle \texttt{CLS} \rangle$ the classification token (detailed later). Each token $c_i \in C$ is represented as the sum of a token embedding, a position embedding, and a segment embedding. These embeddings are then fed into a stack of Transformer encoding blocks (Vaswani et al., 2017), the output of which is used to predict the answer.

In the extractive scenario, we use the output representations corresponding to passage tokens (and $\langle \texttt{CLS} \rangle$ as well) to predict answer boundaries. The traditional answer span extraction strategy in MRC is to have two m-class classifiers separately predict the start and the end of the answer from the passage. Since the softmax function is applied over all tokens in the passage, this strategy can only output one single answer span given a question. As in the event extraction task a sentence can contain multiple event triggers or multiple arguments playing for one particular role, we annotate a BIO tag for each token and adopt a 3-class classifier to predict the tag of each token. BIO tags respectively represent the *beginning(B)*, *inside(I)* and *outside(O)* of an answer span. This strategy allows for outputting multiple answer spans given a passage and a question. The probability of each token c_i being assigned a tag $\in$ *B, I, O* is calculated as:

$$p_i^{tag} = \frac{\exp(\mathbf{w}_{tag}^\top \mathbf{o}_i)}{\sum_{tag'} \exp(\mathbf{w}_{tag'}^\top \mathbf{o}_i)},$$

where $\mathbf{o}_i$ is the BERT output of c_i, and $\mathbf{w}_{tag}$ is trainable parameter. If there is no answer to the question, labels for tokens are all *O*.

In the YES/NO scenario, we use the BERT output corresponding to $\langle \texttt{CLS} \rangle$, i.e., $\mathbf{o}_1$, to conduct the binary classification. The probability of the answer to be YES is calculated as:

$$p_y = \frac{\exp(\mathbf{w}_y^\top \mathbf{o}_1)}{\exp(\mathbf{w}_y^\top \mathbf{o}_1) + \exp(\mathbf{w}_n^\top \mathbf{o}_1)},$$

and likewise for the NO case. Here $\mathbf{w}_y$ and $\mathbf{w}_n$ are also trainable parameters. In both scenarios, we use cross entropy between the prediction and golden labels as our training loss to fine-tune a pre-trained BERT model.

3 Our Approach

MQAEE splits the event extraction task into three sub-tasks: trigger identification, trigger classification, and argument extraction, solved as multi-turn QA in a pipelined fashion. The first turn is devoted to trigger identification, formalized as an extractive MRC problem that recognizes trigger words from a given sentence. The second turn is then designed for trigger classification, formalized as a YES/NO

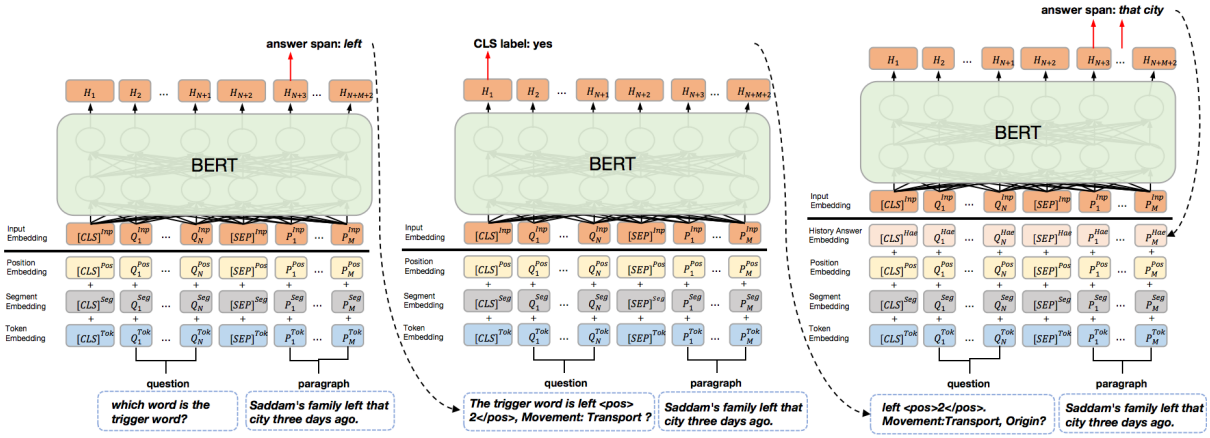

Figure 2: Overall architecture of MQAEE, which performs trigger identification (left), trigger classification (middle), and argument extraction (right) as multi-turn QA in a pipelined fashion. The argument extraction task itself is modeled as a built-in multi-turn QA process.

problem so as to judge whether a recognized trigger belongs to a specific event type. The rest of the QA process deals with argument extraction, which is formalized as successive extractive MRC problems, identifying arguments of specific roles associated with a predicted event type one by one. See Table 1 for an overview of our approach and Figure 2 for the overall architecture.

3.1 Trigger Identification

Given an sentence, trigger identification is to extract a word or phrase that triggers an event. We formulate it as an extractive MRC problem, where the given sentence is taken as a reference passage, and a question is constructed as:

Q: Which word is the trigger word?

By answering this question, we extract contiguous spans from the passage as the triggers. We adopt the BERT-based MRC technique introduced in Section 2.2 to solve this problem. Note that we use the same question for any input sentence in this task.

3.2 Trigger Classification

Trigger classification aims to classify an event to a specific event type based on the identified trigger. We formulate it as a YES/NO MRC problem. The sentence is again taken as a reference passage, and a question is constructed via the following template:

Q: The trigger word is ⟨`trigger`⟩ ⟨`trigger position`⟩, ⟨`event type`⟩, ⟨`argument roles`⟩*?*

Here, ⟨`trigger`⟩ is a previously identified trigger and ⟨`trigger position`⟩ is the corresponding token position in the passage; ⟨`event type`⟩ is a candidate event type to which the trigger will be assigned; ⟨`argument roles`⟩ is a list of argument roles associated with the given type. Take the sentence shown in Figure 1 as an example. The question there will be constructed as *"The trigger word is left ⟨pos⟩ 2 ⟨/pos⟩, Movement:Transport, Agent, Artifact, Vehicle, Time-Within, Origin, Destination, Price?"*, which means *"The trigger word is left, does the event belong to the Movement Transport type, which gets the seven argument roles?"*.[1] The answer YES means the event does belong to the given type, while otherwise the answer NO. We use the BERT-based technique introduced in Section 2.2 to solve this binary classification problem. We pair a sentence with its golden type as a positive training instance, and with any other type as a negative training instance.

Traditional approaches typically adopt sequence labeling techniques to identify triggers and classify them into event types at the same time. Such kind of approaches might perform poorly for event types with only a few training instances, and are not able to generalize to new event types. In contrast, our approach casts trigger extraction as two successive MRC problems. By using the same question in the first and explicitly encoding event types as partial questions in the second, our approach better transfers knowledge from event types with rich training

[1] We omit argument roles from the question in Table 1 and Figure 2 for simplicity.

instances to those with fewer ones, and generalizes well to new event types.

3.3 Argument Extraction

Given an identified trigger of a specific type, argument extraction is to identify arguments and classify them into corresponding roles. Argument extraction is a challenging task. First, arguments are definitely dependent on event types. Different types are supposed to get arguments with different roles. Besides, there could also be dependency among arguments even with the same type. For instance, in the case we have shown in Figure 1, given that "Saddam's family" (the subject of the sentence) is an argument of the role *Artifact*, it is more likely to infer "that city" (the object) might be an argument of the role *Origin*. How to make full use of such complicated dependency stands out as an important factor in argument extraction.

This work formulates argument extraction as a multi-turn QA process, where a series of extractive MRC problems are designed to extract arguments one by one in a pre-defined order (we will describe how to determine the order later). During each turn, we take the sentence as a reference passage, and construct a question with the template:

Q: ⟨trigger⟩ ⟨trigger position⟩. ⟨event type⟩, ⟨argument role⟩*?*

By answering this question, we extract contiguous spans from the passage as the arguments with the specific role, which performs argument identification and classification simultaneously.

This question template naturally models the dependency between arguments and event types. And the dependency among different arguments associated with the same type are modeled by the multi-turn mechanism. To be specific, during each turn, we introduce a *history answer embedding* for each token, indicating whether that token has appeared in any previous answer. If it has, we assign to it a vector embedding $\mathbf{a}$, otherwise another vector embedding $\mathbf{b}$. Such history answer embeddings are shown to be very effective in modeling previous QA history during multi-turn QA (Qu et al., 2019), and would naturally capture the dependency among different arguments in our case. We apply the BERT technique introduced in Section 2.2 to solve the extractive MRC problem during each turn. The difference is that we construct each input representation as a sum of four embeddings (token, position, segment, history answering) rather than the first three, as illustrated in Figure 2.

As far as we know, this is the first work that formulates argument extraction as multi-turn QA. By this new formulation, our approach makes full use of complex dependency among arguments and event types, and generalizes well to new event types with new argument roles.

4 Experiments

This section presents our experiments on the ACE 2005 benchmark,[2] demonstrating the effectiveness and generalization ability of our approach.

4.1 Dataset and Evaluation Metrics

ACE 2005 annotates 8 coarse-grained main event types, 33 event subtypes, and 36 argument role classes. We classify trigger words into the 33 subtypes and use the associated role classes to extract arguments. To make our results directly comparable, we keep the same data split as previous work (Ji and Grishman, 2008; Chen et al., 2015; Liu et al., 2016; Yang and Mitchell, 2016; Nguyen et al., 2016; Sha et al., 2018), which includes 40 newswire documents in the test set, 30 in the development set, and 529 in the training set.

For evaluation, we split the task into four subtasks: trigger identification, trigger classification, argument identification, and argument classification. We follow the criteria of previous work (Chen et al., 2015; Liu et al., 2016; Yang and Mitchell, 2016; Nguyen et al., 2016; Sha et al., 2018) : (1) A trigger is correctly identified iff the predicted trigger span matches with a golden label; (2) A trigger is correctly classified iff it is correctly identified and assigned to the right subtype; (3) An argument is correctly identified iff the subtype is correctly recognized and the predicted argument span matches with a golden label; (4) An argument is correctly classified iff it is correctly identified and the predicted role matches with any of golden labels. We report Precision (P), Recall (R) and F measure (F1) for each of the four sub-tasks.

4.2 Experimental Setups

We compare against the following state-of-the-art methods: (1) **JointBeam** (Li et al., 2013) which jointly extracts event triggers and arguments via structure prediction by well designed features.

[2] https://catalog.ldc.upenn.edu/LDC2006T06

(2) **JointEventEntity** (Yang and Mitchell, 2016) which models the dependencies among events and entities and jointly extracts events and entities.(3) **RBPB** (Sha et al., 2016) which proposes a regularization-based pattern balancing method to extract event triggers and arguments. (4) **dbRNN** (Sha et al., 2018) which adds dependency bridges over Bi-LSTM for event extraction. (5) **DYGIE++** (Wadden et al., 2019) which proposes a multi-task framework for entity, relation and event extraction with contextualized span representations. DYGIE++(ens) indicates the use of 4-model ensemble for trigger detection. All these baselines formulate event extraction as classification tasks, while our approach formulates it as a multi-turn QA task.

We implement our model based on the BERT-Large model same as DYGIE++. To summarize, we maintain three sub-models for MQAEE: a MRC based trigger identifier, a MRC based trigger classifier, a multi-turn MRC based argument extractor. For all the three sub-models, the batch size is set to 8 and the max sequence length is 200. As to the trigger identifier, multi-turn argument extractor, we train the models with a Adam weight decay optimizer with an initial learning rate of 1e-5. The warming up portion for learning rate is 10%. We set the stride in the sliding window for passages to 128, the max question length to 64, and the max answer length to 30. We set the training epoch to 10 for both the trigger identifier and the two argument extractors. Also, a Adam weight decay optimizer with an initial learning rate of 1e-5 is adopted to train the trigger classifier. The warming up portion for learning rate is 10% and the epoch is set to 3.

Model	Tri-Id F1 (%)	Tri-Cls F1 (%)	Arg-Id F1 (%)	Arg-Cls F1 (%)
JointBeam	N/A	64.2	38.0	35.0
RBPB	N/A	67.8	55.4	43.8
JointEventEntity	N/A	68.7	50.6	48.4
dbRNN	N/A	69.6	**57.2**	50.1
DYGIE++	N/A	68.9	54.1	51.4
DYGIE++(ens)	76.5	73.6	55.4	52.5
MQAEE	74.5	71.7	55.2	53.4
MQAEE(ens)	**77.4**	**73.8**	56.7	**55.0**

Table 2: Overall performance compared against state-of-the-art methods. Notations for events are defined as followed: *Tri*: Trigger, *Arg*: Argument, *Id*: Identification, *Cls*: Classification.

4.3 Main Results

Table 2 shows the overall performance of our approach compared against the above state-of-the-art methods on the test dataset.

The results show that our MQAEE outperforms all other models except DYGIE++(ens) on the trigger classification. This is acceptable since DYGIE++(ens) uses 4-model ensemble for trigger detection. Compared to DYGIE++ that is a single model also based on BERT-Large, MQAEE achieves a sharp increase of 2.8% on the *F1* score. Compared to DYGIE++(ens), we adopt the same ensemble setting of DYGIE++(ens) and train our ensemble MQAEE(ens). We can see that MQAEE(ens) has already outperformed DYGIE++(ens) on the trigger identification and trigger classification.

In terms of the results of argument classification, our model achieves the best performance. MQAEE achieves an increase of 2.0% on the *F1* score compared to DYGIE++, and an increase of 0.9% even compared to DYGIE++(ens), which verifies the feasibility and effectiveness of reading comprehension question answering in event extraction. Consistent with that, MQAEE(ens) achieves a sharp increase of 2.5% on the *F* score compared to DYGIE++(ens).

Model	Tri-Id F1 (%)	Tri-Cls F1 (%)	Arg-Id F1 (%)	Arg-Cls F1 (%)
QAEE	74.5	71.7	52.4	50.4
MQAEE (rnd)	74.5	71.7	53.9	51.8
MQAEE (-tri)	74.5	71.7	53.7	51.1
MQAEE	74.5	71.7	**55.2**	**53.4**

Table 3: Results of different settings of MQAEE.

4.4 Effect of Multi-turn Question Answering

In order to evaluate the effectiveness of multi-turn QA, we evaluate our approach in four settings: (1) **QAEE** that employs a single-turn QA mechanism for argument extraction. In QAEE, arguments are extracted independently, each by solving a separate extractive MRC problem. (2) **MQAEE(-tri)**, (3)**MQAEE (rnd)** and (4) **MQAEE** that apply multi-turn QA mechanisms for argument extraction. In these three settings, arguments are extracted successively one by one. The extraction of an argument is dependent upon the extraction of previous ones, by which the dependency among different arguments will be better captured. MQAEE(-tri) denotes questions are constructed without consideration of extracted triggers and trigger positions during argument extraction. MQAEE (rnd) determines the extraction order by randomly shuffling

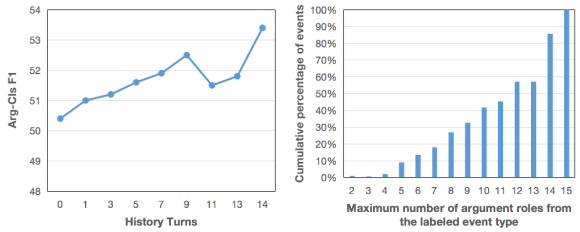

Figure 3: Effect of QA history turns, the right part is cumulative distribution of events over max number of argument roles.

argument roles, while MQAEE determines the order by ranking argument roles according to their classification precision of QAEE on the development set. All the four settings use the same procedure for trigger identification and classification, as illustrated in Figure 2 (left and middle).

As showed in Table 3, comparing to QAEE, MQAEE can achieve a better result with improvement of 3.0% on the *F1* score, which shows that the introduction of multi-turn question answering mechanism can indeed bring information gain and improvement. Comparing to MQAEE (rnd), MQAEE achieve better performance, which indicates that a better extraction order is helpful for the multi-turn extraction. Without considering the extracted trigger, MQAEE(-tri) have a 2.3% degradation, which verifies the importance of trigger information in argument extraction. Also, the same observation appears in argument identification.

4.5 Effect of QA History Turns

We present an analysis on the effect of different QA history turns of MQAEE. When the history turn is set to N, it means only the N previous history QA turns are considered during each turn. If previous QA history turns is less than N, the model keeps all QA history. As showed in the right part of Figure 3, events whose event types containing a maximum argument roles number greater than 14 have a large proportion. As showed in Figure 3, with the increase of history turns, MQAEE achieves better performance, and achieves the best performance with 14 history turns(maximum number of argument roles of event types is 15), which demonstrates that the history answer embedding can model complicated QA history and more QA history turns indeed brings some gain. And it is verified that the trend of argument classification performance is related to the maximum number of argument roles of events.

4.6 Case Study

Table 4 shows the event extraction results conducted by MQAEE and QAEE models. In this sentence, the trigger word is *"appointed"*. Both MQAEE and QAEE correctly extracts the arguments of roles *"Person, Time-Within"* and MQAEE can extract more arguments than QAEE, by correctly extracting the arguments of roles *"Entity, Position"*. As the multi-turn argument extraction is conducted in this order of argument roles: *"Time-Within, Person, Entity, Position"*, when knowing the subject *"Diller"* playing the role *"Person"* that means the employee in this scenario, our model can predicts *"Vivendi Universal's U.S.-based entertainment assets"* as the argument playing the role *"Entity"* that means the employer. The phrase *"interim CEO of Vivendi Universal's U.S.-based entertainment assets"* is the object of the sentence, which is syntactically strongly related with the argument *"Diller"* (the subject). Thus, the probability of the object being an argument should be increased. Examples in Table 4 indicates that the multi-turn question answering mechanism can make use of the association between arguments and improve the recall of argument extraction. This phenomenon of MQAEE is in consistency with the increase of 3.0% on the *F1* score of argument classification compared to QAEE.

4.7 Generalization Ability

In order to verify the generalization ability of MQAEE, we conduct a few-shot learning experiment. In few-shot learning, the terminology "N-way K-shot classification" denotes training the classification model with training dataset containing N classes and K labeled samples per class, and evaluating the model on test dataset of the same N class.

The basic idea of evaluating generalization ability of our model is: (1) train our model using all samples of Top M most popular event types in the training dataset and development dataset to acquire prior knowledge. (2) finetune the model using few samples of the remaining N event types in the training dataset and development dataset. (3) evaluate the results in the remaining N event types in the test dataset. To verify the effect of the number of event types used in step (1), we set M as 5, 10, 20. As the remaining N event types ought be different from the top M event types, we set N as 13 and we adopt the settings of 13-way with 1-shot and 5-shot.

Sentence	Diller was appointed interim CEO of Vivendi Universal's U.S.-based entertainment assets last year.			
Trigger	appointed			
Event Type	Personnel:Start-Position			
Argument Role	Person	Entity	Position	Time-Within
Golden	Diller	Vivendi Universal's U.S.-based entertainment assets	interim CEO of Vivendi Universal's U.S.-based entertainment assets	last year
QAEE	Diller	Vivendi Universal	NULL	last year
MQAEE	Diller	*Vivendi Universal's U.S.-based entertainment assets*	*interim CEO of Vivendi Universal's U.S.-based entertainment assets*	*last year*

Table 4: Case study of MQAEE and QAEE

Setting	Tri-Id	Tri-Cls	Arg-Id	Arg-Cls
main	76.9	76.9	61.1	61.1
M=5 5-shot	43.1	43.1	37.8	37.8
M=10 5-shot	51.5	51.5	43.1	42.5
M=20 5-shot	57.1	55.6	49.3	47.7
M=5 1-shot	40.7	40.0	15.7	15.5
M=10 1-shot	40.9	40.0	33.7	33.4
M=20 1-shot	46.4	45.6	38.1	38.1

Table 5: 13-way few-shot learning performance. "main" denotes MQAEE trained in section 4.3 and evaluated in the same 13 event types.

We conduct six groups of experiments for MQAEE corresponding to the six kinds of settings. To verify the performance of the few-shot learning, we evaluate the MQAEE model trained in section 4.3 on the same 13 event types for comparison denoted as "main". We use the same hyper-parameters as that of the above experiment.

Table 5 shows the result of the generalization experiment. We can see that with the increase in the number of event types used for training, our model achieves better performance. when trained with the top 20 event types and finetuned in 13-way with 5-shot, MQAEE achieves competitive results on the remaining 13 event types comparing to our MQAEE trained with all 33 event types, which demonstrates good generalization capability of MQAEE.

5 Related Work

Machine Reading Comprehension. Machine reading comprehension is a basic task of textual question answering, which makes rapid progress in recent years. Mainstream approaches (Seo et al., 2016; Wang and Jiang, 2016; Xiong et al., 2018; Joshi et al., 2017; Dunn et al., 2017; Shen et al., 2016; Wang et al., 2017a,b; Tan et al., 2017) formulate reading comprehension as extracting answer spans from a given passage. Generally, answer spans extraction is conducted by predicting the starting and the ending positions of the answers. The recent rapid development of pre-trained language models such as ELMo (Peters et al., 2018) or BERT (Devlin et al., 2019) has achieved significant improvements on downstream NLP tasks, and the pre-trained models have been verified to be extraordinarily beneficial for the MRC tasks like SQUAD (Rajpurkar et al., 2016).

Recently, there has been a trend of formulating non-QA NLP tasks as QA-based ones. Levy et al. (2017) and McCann et al. (2018) tried to formulate relation extraction as single-turn QA tasks. Li et al. (2019) later introduced a multi-turn QA mechanism to further model hierarchical tag dependency for the task. Our work considers the more challenging event extraction task, with much more complicated tag interactions and dependency, particularly suitable for a multi-turn QA infrastructure.

6 Conclusion and Future Work

This work presents a novel multi-turn QA paradigm for event extraction, referred to as MQAEE. It splits event extraction into three sub-tasks: trigger identification, trigger classification, and argument extraction, solved as a series of reading comprehension problems in a pipelined fashion. Within the multi-turn process, a history answer embedding strategy is further introduced to effectively model QA history. By this new formulation, MQAEE makes full use of dependency among arguments and event types, and generalizes well to new types with new argument roles. Experimental results on ACE 2005 demonstrate the effectiveness and generalization ability of our approach. As future work, We would like to apply reinforcement learning to determine a better QA order for argument extraction in the multi-turn framework, and explore more variants of the model architecture.

References

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

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

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, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Matthew Dunn, Levent Sagun, Mike Higgins, V. Ugur Güney, Volkan Cirik, and Kyunghyun Cho. 2017. Searchqa: A new q&a dataset augmented with context from a search engine. *CoRR*, abs/1704.05179.

Ralph Grishman, David Westbrook, and Adam Meyers. 2005. Nyu's english ace 2005 system description. *ACE*, 5.

Yu Hong, Jianfeng Zhang, Bin Ma, Jianmin Yao, Guodong Zhou, and Qiaoming Zhu. 2011. Using cross-entity inference to improve event extraction. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1127–1136, Portland, Oregon, USA. Association for Computational Linguistics.

Lifu Huang, Heng Ji, Kyunghyun Cho, Ido Dagan, Sebastian Riedel, and Clare Voss. 2018. Zero-shot transfer learning for event extraction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2160–2170, Melbourne, Australia. Association for Computational Linguistics.

Heng Ji and Ralph Grishman. 2008. Refining event extraction through cross-document inference. In *Proceedings of ACL-08: HLT*, pages 254–262, Columbus, Ohio. Association for Computational Linguistics.

Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. *CoRR*, abs/1705.03551.

Omer Levy, Minjoon Seo, Eunsol Choi, and Luke Zettlemoyer. 2017. Zero-shot relation extraction via reading comprehension. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 333–342, Vancouver, Canada. Association for Computational Linguistics.

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 (Volume 1: Long Papers)*, pages 73–82, Sofia, Bulgaria. Association for Computational Linguistics.

Xiaoya Li, Fan Yin, Zijun Sun, Xiayu Li, Arianna Yuan, Duo Chai, Mingxin Zhou, and Jiwei Li. 2019. Entity-relation extraction as multi-turn question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1340–1350, Florence, Italy. Association for Computational Linguistics.

Shasha Liao and Ralph Grishman. 2010. Using document level cross-event inference to improve event extraction. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 789–797, Uppsala, Sweden. Association for Computational Linguistics.

Shulin Liu, Kang Liu, Shizhu He, and Jun Zhao. 2016. A probabilistic soft logic based approach to exploiting latent and global information in event classification. In *AAAI*.

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

Wei Lu and Dan Roth. 2012. Automatic event extraction with structured preference modeling. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 835–844, Jeju Island, Korea. Association for Computational Linguistics.

Bryan McCann, Nitish Shirish Keskar, Caiming Xiong, and Richard Socher. 2018. The natural language decathlon: Multitask learning as question answering. *CoRR*, abs/1806.08730.

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

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke

Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Chen Qu, Liu Yang, Minghui Qiu, W. Bruce Croft, Yongfeng Zhang, and Mohit Iyyer. 2019. Bert with history answer embedding for conversational question answering. In *SIGIR*.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2016. Bidirectional attention flow for machine comprehension. *ArXiv*, abs/1611.01603.

Lei Sha, Jing Liu, Chin-Yew Lin, Sujian Li, Baobao Chang, and Zhifang Sui. 2016. RBPB: Regularization-based pattern balancing method for event extraction. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1224–1234, Berlin, Germany. Association for Computational Linguistics.

Lei Sha, Feng Qian, Baobao Chang, and Zhifang Sui. 2018. Jointly extracting event triggers and arguments by dependency-bridge rnn and tensor-based argument interaction. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Yelong Shen, Po-Sen Huang, Jianfeng Gao, and Weizhu Chen. 2016. Reasonet: Learning to stop reading in machine comprehension. *CoRR*, abs/1609.05284.

Chuanqi Tan, Furu Wei, Nan Yang, Weifeng Lv, and Ming Zhou. 2017. S-net: From answer extraction to answer generation for machine reading comprehension. *CoRR*, abs/1706.04815.

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*.

David Wadden, Ulme Wennberg, Yi Luan, and Hannaneh Hajishirzi. 2019. Entity, relation, and event extraction with contextualized span representations.

Chuan Wang, Nianwen Xue, and Sameer Pradhan. 2015. Boosting transition-based amr parsing with refined actions and auxiliary analyzers. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 857–862.

Shuohang Wang and Jing Jiang. 2016. Machine comprehension using match-lstm and answer pointer. *CoRR*, abs/1608.07905.

Shuohang Wang, Mo Yu, Jing Jiang, Wei Zhang, Xiaoxiao Guo, Shiyu Chang, Zhiguo Wang, Tim Klinger, Gerald Tesauro, and Murray Campbell. 2017a. Evidence aggregation for answer re-ranking in open-domain question answering. *CoRR*, abs/1711.05116.

Wenhui Wang, Nan Yang, Furu Wei, Baobao Chang, and Ming Zhou. 2017b. Gated self-matching networks for reading comprehension and question answering. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 189–198, Vancouver, Canada. Association for Computational Linguistics.

Caiming Xiong, Victor Zhong, and Richard Socher. 2018. DCN+: Mixed objective and deep residual coattention for question answering. In *International Conference on Learning Representations*.

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

Sen Yang, Dawei Feng, Linbo Qiao, Zhigang Kan, and Dongsheng Li. 2019. Exploring pre-trained language models for event extraction and generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5284–5294, Florence, Italy. Association for Computational Linguistics.