CLEVE: Contrastive Pre-training for Event Extraction

Ziqi Wang[1]*, **Xiaozhi Wang**[1]*, **Xu Han**[1], **Yankai Lin**[3], **Lei Hou**[1,2]† ,
Zhiyuan Liu[1,2], **Peng Li**[3], **Juanzi Li**[1,2], **Jie Zhou**[3]
[1]Department of Computer Science and Technology, BNRist;
[2]KIRC, Institute for Artificial Intelligence,
Tsinghua University, Beijing, 100084, China
[3]Pattern Recognition Center, WeChat AI, Tencent Inc, China
{ziqi-wan16, wangxz20, hanxu17}@mails.tsinghua.edu.cn

Abstract

Event extraction (EE) has considerably benefited from pre-trained language models (PLMs) by fine-tuning. However, existing pre-training methods have not involved modeling event characteristics, resulting in the developed EE models cannot take full advantage of large-scale unsupervised data. To this end, we propose CLEVE, a contrastive pre-training framework for EE to better learn event knowledge from large unsupervised data and their semantic structures (e.g. AMR) obtained with automatic parsers. CLEVE contains a text encoder to learn event semantics and a graph encoder to learn event structures respectively. Specifically, the text encoder learns event semantic representations by self-supervised contrastive learning to represent the words of the same events closer than those unrelated words; the graph encoder learns event structure representations by graph contrastive pre-training on parsed event-related semantic structures. The two complementary representations then work together to improve both the conventional supervised EE and the unsupervised "liberal" EE, which requires jointly extracting events and discovering event schemata without any annotated data. Experiments on ACE 2005 and MAVEN datasets show that CLEVE achieves significant improvements, especially in the challenging unsupervised setting. The source code and pre-trained checkpoints can be obtained from https://github.com/THU-KEG/CLEVE.

1 Introduction

Event extraction (EE) is a long-standing crucial information extraction task, which aims at extracting event structures from unstructured text. As illustrated in Figure 1, it contains event detection task to identify event triggers (the word "attack") and
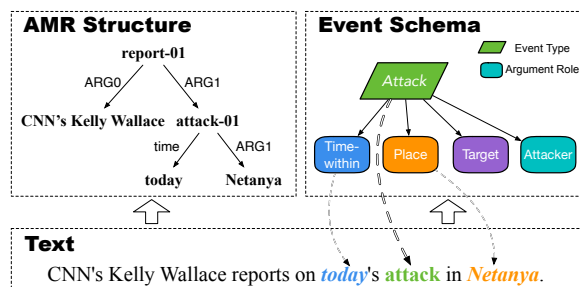

Figure 1: An example sampled from the ACE 2005 dataset with its event annotation and AMR structure.

classify event types (Attack), as well as event argument extraction task to identify entities serving as event arguments ("today" and "Netanya") and classify their argument roles (Time-within and Place) (Ahn, 2006). By explicitly capturing the event structure in the text, EE can benefit various downstream tasks such as information retrieval (Glavaš and Šnajder, 2014) and knowledge base population (Ji and Grishman, 2011).

Existing EE methods mainly follow the supervised-learning paradigm to train advanced neural networks (Chen et al., 2015; Nguyen et al., 2016; Nguyen and Grishman, 2018) with human-annotated datasets and pre-defined event schemata. These methods work well in lots of public benchmarks such as ACE 2005 (Walker et al., 2006) and TAC KBP (Ellis et al., 2016), yet they still suffer from data scarcity and limited generalizability. Since annotating event data and defining event schemata are especially expensive and labor-intensive, existing EE datasets typically only contain thousands of instances and cover limited event types. Thus they are inadequate to train large neural models (Wang et al., 2020) and develop methods that can generalize to continually-emerging new event types (Huang and Ji, 2020).

Inspired by the success of recent pre-trained language models (PLMs) for NLP tasks, some pio-

---

* indicates equal contribution
† Correspondence to L.Hou (houlei@tsinghua.edu.cn)

neering work (Wang et al., 2019a; Wadden et al., 2019) attempts to fine-tune general PLMs (e.g, BERT (Devlin et al., 2019)) for EE. Benefiting from the strong general language understanding ability learnt from large-scale unsupervised data, these PLM-based methods have achieved state-of-the-art performance in various public benchmarks.

Although leveraging unsupervised data with pre-training has gradually become a consensus for EE and NLP community, there still lacks a pre-training method orienting event modeling to take full advantage of rich event knowledge lying in large-scale unsupervised data. The key challenge here is to find reasonable self-supervised signals (Chen et al., 2017; Wang et al., 2019a) for the diverse semantics and complex structures of events. Fortunately, previous work (Aguilar et al., 2014; Huang et al., 2016) has suggested that sentence semantic structures, such as abstract meaning representation (AMR) (Banarescu et al., 2013), contain broad and diverse semantic and structure information relating to events. As shown in Figure 1, the parsed AMR structure covers not only the annotated event (`Attack`) but also the event that is not defined in the ACE 2005 schema (`Report`).

Considering the fact that the AMR structures of large-scale unsupervised data can be easily obtained with automatic parsers (Wang et al., 2015), we propose CLEVE, an event-oriented contrastive pre-training framework utilizing AMR structures to build self-supervision signals. CLEVE consists of two components, including a text encoder to learn event semantics and a graph encoder to learn event structure information. Specifically, to learn effective event semantic representations, we employ a PLM as the text encoder and encourage the representations of the word pairs connected by the `ARG`, `time`, `location` edges in AMR structures to be closer in the semantic space than other unrelated words, since these pairs usually refer to the trigger-argument pairs of the same events (as shown in Figure 1) (Huang et al., 2016). This is done by contrastive learning with the connected word pairs as positive samples and unrelated words as negative samples. Moreover, considering event structures are also helpful in extracting events (Lai et al., 2020) and generalizing to new event schemata (Huang et al., 2018), we need to learn transferable event structure representations. Hence we further introduce a graph neural network (GNN) as the graph encoder to encode AMR

structures as structure representations. The graph encoder is contrastively pre-trained on the parsed AMR structures of large unsupervised corpora with AMR subgraph discrimination as the objective.

By fine-tuning the two pre-trained models on downstream EE datasets and jointly using the two representations, CLEVE can benefit the conventional supervised EE suffering from data scarcity. Meanwhile, the pre-trained representations can also directly help extract events and discover new event schemata without any known event schema or annotated instances, leading to better generalizability. This is a challenging unsupervised setting named "liberal event extraction" (Huang et al., 2016). Experiments on the widely-used ACE 2005 and the large MAVEN datasets indicate that CLEVE can achieve significant improvements in both settings.

2 Related Work

Event Extraction. Most of the existing EE works follow the supervised learning paradigm. Traditional EE methods (Ji and Grishman, 2008; Gupta and Ji, 2009; Li et al., 2013) rely on manually-crafted features to extract events. In recent years, the neural models become mainstream, which automatically learn effective features with neural networks, including convolutional neural networks (Nguyen and Grishman, 2015; Chen et al., 2015), recurrent neural networks (Nguyen et al., 2016), graph convolutional networks (Nguyen and Grishman, 2018; Lai et al., 2020). With the recent successes of BERT (Devlin et al., 2019), PLMs have also been used for EE (Wang et al., 2019a,b; Yang et al., 2019; Wadden et al., 2019; Tong et al., 2020). Although achieving remarkable performance in benchmarks such as ACE 2005 (Walker et al., 2006) and similar datasets (Ellis et al., 2015, 2016; Getman et al., 2017; Wang et al., 2020), these PLM-based works solely focus on better fine-tuning rather than pre-training for EE. In this paper, we study pre-training to better utilize rich event knowledge in large-scale unsupervised data.

Event Schema Induction. Supervised EE models cannot generalize to continually-emerging new event types and argument roles. To this end, Chambers and Jurafsky (2011) explore to induce event schema from raw text by unsupervised clustering. Following works introduce more features like coreference chains (Chambers, 2013) and entities (Nguyen et al., 2015; Sha et al., 2016). Recently, Huang and Ji (2020) move to the semi-
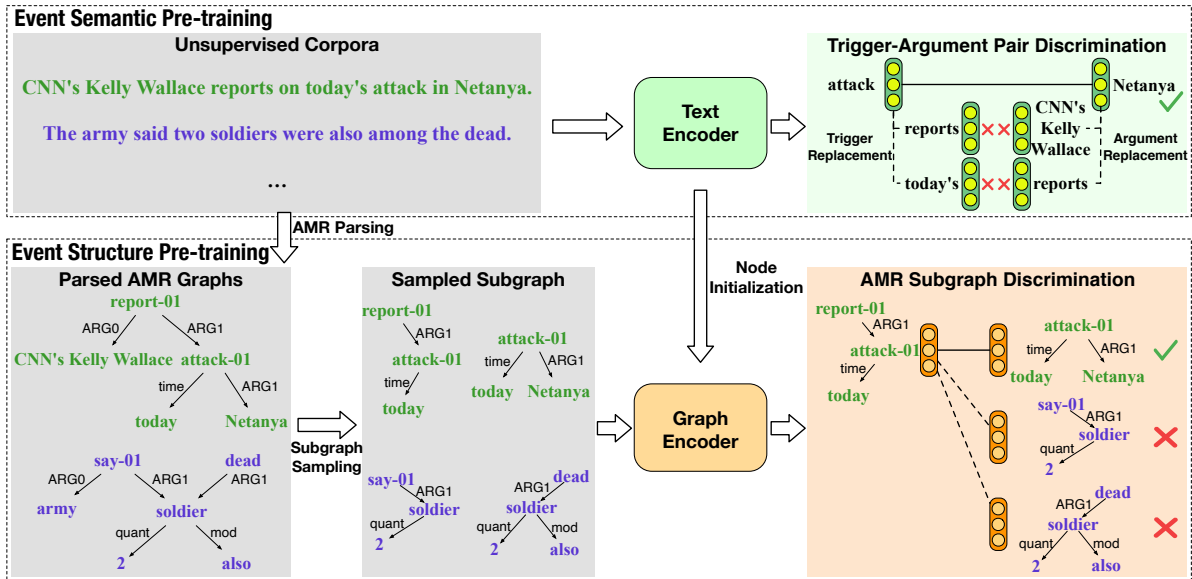
Figure 2: Overall CLEVE framework. Best viewed in color.

supervised setting allowing to use annotated data of known types. Following Huang et al. (2016), we evaluate the generalizability of CLEVE in the most challenging unsupervised "liberal" setting, which requires to induce event schemata and extract event instances only from raw text at the same time.

Contrastive Learning. Contrastive learning was initiated by Hadsell et al. (2006) following an intuitive motivation to learn similar representations for "neighboors" and distinct representations for "non-neighboors", and is further widely used for self-supervised representation learning in various domains, such as computer vision (Wu et al., 2018; Oord et al., 2018; Hjelm et al., 2019; Chen et al., 2020; He et al., 2020) and graph (Qiu et al., 2020; You et al., 2020; Zhu et al., 2020). In the context of NLP, many established representation learning works can be viewed as contrastive learning methods, such as Word2Vec (Mikolov et al., 2013), BERT (Devlin et al., 2019; Kong et al., 2020) and ELECTRA (Clark et al., 2020). Similar to this work, contrastive learning is also widely-used to help specific tasks, including question answering (Yeh and Chen, 2019), discourse modeling (Iter et al., 2020), natural language inference (Cui et al., 2020) and relation extraction (Peng et al., 2020).

3 Methodology

The overall CLEVE framework is illustrated in Figure 2. As shown in the illustration, our contrastive pre-training framework CLEVE consists of two components: event semantic pre-training and event

structure pre-training, of which details are introduced in Section 3.2 and Section 3.3, respectively. At the beginning of this section, we first introduce the required preprocessing in Section 3.1, including the AMR parsing and how we modify the parsed AMR structures for our pre-training.

3.1 Preprocessing

CLEVE relies on AMR structures (Banarescu et al., 2013) to build broad and diverse self-supervision signals for learning event knowledge from large-scale unsupervised corpora. To do this, we use automatic AMR parsers (Wang et al., 2015; Xu et al., 2020) to parse the sentences in unsupervised corpora into AMR structures. Each AMR structure is a directed acyclic graph with concepts as nodes and semantic relations as edges. Moreover, each node typically only corresponds to at most one word, and a multi-word entity will be represented as a list of nodes connected with `name` and `op` (conjunction operator) edges. Considering pre-training entity representations will naturally benefits event argument extraction, we merge these lists into single nodes representing multi-word entities (like the "CNN's Kelly Wallace" in Figure 1) during both event semantic and structure pre-training. Formally, given a sentence s in unsupervised corpora, we obtain its AMR graph $g_s = (V_s, E_s)$ after AMR parsing, where V_s is the node set after word merging and E_s denotes the edge set. $E_s = \{(u, v, r) \mid (u, v) \in V_s \times V_s, r \in \mathcal{R}\}$, where $\mathcal{R}$ is the set of defined semantic relation types.

3.2 Event Semantic Pre-training

To model diverse event semantics in large unsupervised corpora and learn contextualized event semantic representations, we adopt a PLM as the text encoder and train it with the objective to discriminate various trigger-argument pairs.

Text Encoder

Like most PLMs, we adopt a multi-layer Transformer (Vaswani et al., 2017) as the text encoder since its strong representation capacity. Given a sentence $s = \{w_1, w_2, \ldots, w_n\}$ containing n tokens, we feed it into the multi-layer Transformer and use the last layer's hidden vectors as token representations. Moreover, a node $v \in V_s$ may correspond to a multi-token text span in s and we need a unified representation for the node in pre-training. As suggested by Baldini Soares et al. (2019), we insert two special markers [E1] and [/E1] at the beginning and ending of the span, respectively. Then we use the hidden vector for [E1] as the span representation $\mathbf{x}_v$ of the node v. And we use different marker pairs for different nodes.

As our event semantic pre-training focuses on modeling event semantics, we start our pre-training from a well-trained general PLM to obtain general language understanding abilities. CLEVE is agnostic to the model architecture and can use any general PLM, like BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019).

Trigger-Argument Pair Discrimination

We design trigger-argument pair discrimination as our contrastive pre-training task for event semantic pre-training. The basic idea is to learn closer representations for the words in the same events than the unrelated words. We note that the words connected by ARG, time and location edges in AMR structures are quite similar to the trigger-argument pairs in events (Huang et al., 2016, 2018), i.e., the key words evoking events and the entities participating events. For example, in Figure 1, "Netanya" is an argument for the "attack" event, while the disconnected "CNN's Kelly Wallace" is not. With this observation, we can use these special word pairs as positive trigger-argument samples and train the text encoder to discriminate them from negative samples, so that the encoder can learn to model event semantics without human annotation.

Let $\mathcal{R}_p = \{\text{ARG}, \text{time}, \text{location}\}$ and $P_s = \{(u,v)|\exists(u,v,r) \in E_s, r \in \mathcal{R}_p\}$ denotes the set of positive trigger-argument pairs in sentence s.

For a specific positive pair $(t, a) \in P_s$, as shown in Figure 2, we construct its corresponding negative samples with trigger replacement and argument replacement. Specifically, in the trigger replacement, we construct m_t number of negative pairs by randomly sample m_t number of negative triggers $\hat{t} \in V_s$ and combine them with the positive argument a. A negative trigger $\hat{t}$ must do not have a directed ARG, time or location edge with a, i.e., $\nexists(\hat{t}, a, r) \in E_s, r \in \mathcal{R}_p$. Similarly, we construct m_a more negative pairs by randomly sample m_a number of negative arguments $\hat{a} \in V_s$ satisfying $\nexists(t, \hat{a}, r) \in E_s, r \in \mathcal{R}_p$. As the example in Figure 2, ("attack", "reports") is a valid negative sample for the positive sample ("attack", "Netanya"), but ("attack", "today's") is not valid since there is a ("attack", "today's", time) edge.

To learn to discriminate the positive trigger-argument pair from the negative pairs and so that model event semantics, we define the training objective for a positive pair (t, a) as a cross-entropy loss of classifying the positive pair correctly:

$$
\begin{aligned}
\mathcal{L}_{t,a} = & -\mathbf{x}_t^\top W \mathbf{x}_a \\
& + \log\Big(\exp\big(\mathbf{x}_t^\top W \mathbf{x}_a\big) + \sum_{i=1}^{m_t} \exp\big(\mathbf{x}_{\hat{t}_i}^\top W \mathbf{x}_a\big) \\
& + \sum_{j=1}^{m_a} \exp\big(\mathbf{x}_t^\top W \mathbf{x}_{\hat{a}_j}\big)\Big),
\end{aligned}
\tag{1}
$$

where m_t, m_a are hyper-parameters for negative sampling, and W is a trainable matrix learning the similarity metric. We adopt the cross-entropy loss here since it is more effective than other contrastive loss forms (Oord et al., 2018; Chen et al., 2020).

Then we obtain the overall training objective for event semantic pre-training by summing up the losses of all the positive pairs of all sentences s in the mini batch B_s:

$$
\mathcal{L}_{sem}(\theta) = \sum_{s \in B_s} \sum_{(t,a) \in P_s} \mathcal{L}_{t,a},
\tag{2}
$$

where θ denotes the trainable parameters, including the text encoder and W.

3.3 Event Structure Pre-training

Previous work has shown that event-related structures are helpful in extracting new events (Lai et al., 2020) as well as discovering and generalizing to new event schemata (Huang et al., 2016, 2018; Huang and Ji, 2020). Hence we conduct event structure pre-training on a GNN as graph encoder to learn transferable event-related structure representations with recent advances in graph contrastive

pre-training (Qiu et al., 2020; You et al., 2020; Zhu et al., 2020). Specifically, we pre-train the graph encoder with AMR subgraph discrimination task.

Graph Encoder

In CLEVE, we utilize a GNN to encode the AMR (sub)graph to extract the event structure information of the text. Given a graph g, the graph encoder represents it with an graph embedding $\mathbf{g} = \mathcal{G}(g, \{\mathbf{x}_v\})$, where $\mathcal{G}(\cdot)$ is the graph encoder and $\{\mathbf{x}_v\}$ denotes the initial node representations fed into the graph encoder. CLEVE is agnostic to specific model architectures of the graph encoder. Here we use a state-of-the-art GNN model, Graph Isomorphism Network (Xu et al., 2019), as our graph encoder for its strong representation ability.

We use the corresponding text span representations $\{\mathbf{x}_v\}$ produced by our pre-trained text encoder (introduced in Section 3.2) as the initial node representations for both pre-training and inference of the graph encoder. This node initialization also implicitly aligns the semantic spaces of event semantic and structure representations in CLEVE, so that can make them cooperate better.

AMR Subgraph Discrimination

To learn transferable event structure representations, we design the AMR subgraph discrimination task for event structure pre-training. The basic idea is to learn similar representations for the subgraphs sampled from the same AMR graph by discriminating them from subgraphs sampled from other AMR graphs (Qiu et al., 2020).

Given a batch of m AMR graphs $\{g_1, g_2, \ldots, g_m\}$, each graph corresponds to a sentence in unsupervised corpora. For the i-th graph g_i, we randomly sample two subgraphs from it to get a positive pair a_{2i-1} and a_{2i}. And all the subgraphs sampled from the other AMR graphs in the mini-batch serve as negative samples. Like in Figure 2, the two green (w/ "attack") subgraphs are a positive pair while the other two subgraphs sampled from the purple (w/ "solider") graph are negative samples. Here we use the subgraph sampling strategy introduced by Qiu et al. (2020), whose details are shown in Appendix C.

Similar to event semantic pre-training, we adopt the graph encoder to represent the samples $\mathbf{a}_i = \mathcal{G}(a_i, \mathbf{x}_v)$ and define the training objective as:

$$\mathcal{L}_{str}(\theta) = -\sum_{i=1}^{m} \log \frac{\exp\left(\mathbf{a}_{2i-1}^{\top}\mathbf{a}_{2i}\right)}{\sum_{j=1}^{2m} \mathbf{1}_{[j \neq 2i-1]} \exp\left(\mathbf{a}_{2i-1}^{\top}\mathbf{a}_j\right)}, \quad (3)$$

where $\mathbf{1}_{[j \neq 2i-1]} \in \{0, 1\}$ is an indicator function evaluating to 1 iff $j \neq 2i - 1$ and θ is the trainable parameters of graph encoder.

4 Experiment

We evaluate our methods in both the supervised setting and unsupervised "liberal" setting of EE.

4.1 Pre-training Setup

Before the detailed experiments, we introduce the pre-training setup of CLEVE in implementation. We adopt the New York Times Corpus (NYT)[1] (Sandhaus, 2008) as the unsupervised pre-training corpora for CLEVE. It contains over 1.8 million articles written and published by the New York Times between January 1, 1987, and June 19, 2007. We only use its raw text and obtain the AMR structures with a state-of-the-art AMR parser (Xu et al., 2020). We choose NYT corpus because (1) it is large and diverse, covering a wide range of event semantics, and (2) its text domain is similar to our principal evaluation dataset ACE 2005, which is helpful (Gururangan et al., 2020). To prevent data leakage, we remove all the articles shown up in ACE 2005 from the NYT corpus during pre-training. Moreover, we also study the effect of different AMR parsers and pre-training corpora in Section 5.2 and Section 5.3, respectively.

For the text encoder, we use the same model architecture as RoBERTa (Liu et al., 2019), which is with 24 layers, 1024 hidden dimensions and 16 attention heads, and we start our event semantic pre-training from the released checkpoint[2]. For the graph encoder, we adopt a graph isomorphism network (Xu et al., 2019) with 5 layers and 64 hidden dimensions, and pre-train it from scratch. For the detailed hyperparameters for pre-training and fine-tuning, please refer to Appendix D.

4.2 Adaptation of CLEVE

As our work focuses on pre-training rather than fine-tuning for EE, we use straightforward and common techniques to adapt pre-trained CLEVE to downstream EE tasks. In the supervised setting, we adopt dynamic multi-pooling mechanism (Chen et al., 2015; Wang et al., 2019a,b) for the text encoder and encode the corresponding local subgraphs with the graph encoder. Then we concate-

---

[1] https://catalog.ldc.upenn.edu/LDC2008T19
[2] https://github.com/pytorch/fairseq

Metric	ED			EAE		
	P	**R**	**F1**	**P**	**R**	**F1**
JointBeam	73.7	62.3	67.5	64.7	44.4	52.7
DMCNN	75.6	63.6	69.1	62.2	46.9	53.5
dbRNN	74.1	69.8	71.9	66.2	52.8	58.7
GatedGCN	**78.8**	76.3	77.6	—	—	—
SemSynGTN	—	—	—	**69.3**	55.9	61.9
RCEE_ER	75.6	74.2	74.9	63.0	64.2	**63.6**
RoBERTa	75.1	79.2	77.1	53.5	66.8	59.4
CLEVE	78.1	**81.5**	**79.8**	55.4	**68.0**	61.1
w/o semantic	75.3	79.7	77.4	53.8	67.0	59.7
w/o structure	78.0	81.1	79.5	55.1	67.6	60.7
on ACE (golden)	76.2	79.8	78.0	54.2	67.5	60.1
on ACE (AMR)	75.7	79.5	77.6	53.6	66.9	59.5

Table 1: Supervised EE performance (%) of various models on ACE 2005.

Metric	ED		
	P	**R**	**F1**
DMCNN	**66.3**	55.9	60.6
BiLSTM	59.8	67.0	62.8
BiLSTM+CRF	63.4	64.8	64.1
MOGANED	63.4	64.1	63.8
DMBERT	62.7	72.3	67.1
BERT+CRF	65.0	70.9	67.8
RoBERTa	64.3	72.2	68.0
CLEVE	64.9	**72.6**	**68.5**
w/o semantic	64.5	72.4	68.2
w/o structure	64.7	72.5	68.4

Table 2: Supervised EE performance (%) of various models on MAVEN.

nate the two representations as features and fine-tune CLEVE on supervised datasets. In the unsupervised "liberal" setting, we follow the overall pipeline of Huang et al. (2016) and directly use the representations produced by pre-trained CLEVE as the required trigger/argument semantic representations and event structure representations. For the details, please refer to Appendix A.

4.3 Supervised EE

Dataset and Evaluation

We evaluate our models on the most widely-used ACE 2005 English subset (Walker et al., 2006) and the newly-constructed large-scale MAVEN (Wang et al., 2020) dataset. ACE 2005 contains 599 English documents, which are annotated with 8 event types, 33 subtypes, and 35 argument roles. MAVEN contains 4, 480 documents and 168 event types, which can only evaluate event detection. We split ACE 2005 following previous EE work (Liao and Grishman, 2010; Li et al., 2013; Chen et al., 2015) and use the official split for MAVEN. EE performance is evaluated with the performance of two subtasks: Event Detection (ED) and Event Argument Extraction (EAE). We report the precision (P), recall (R) and F1 scores as evaluation results, among which F1 is the most comprehensive metric.

Baselines We fine-tune our pre-trained **CLEVE** and set the original **RoBERTa** without our event semantic pre-training as an important baseline. To do ablation studies, we evaluate two variants of CLEVE on both datasets: the **w/o semantic** model adopts a vanilla RoBERTa without event semantic pre-training as the text encoder, and the **w/o structure** only uses the event semantic representations

without event structure pre-training.

On ACE 2005, we set two more variants to investigate the effectiveness of CLEVE. The **on ACE (golden)** model is pre-trained with the golden trigger-argument pairs and event structures of ACE 2005 training set instead of the AMR structures of NYT. Similarly, the **on ACE (AMR)** model is pre-trained with the parsed AMR structures of ACE 2005 training set. We also compare CLEVE with various baselines, including: (1) feature-based method, the top-performing **JointBeam** (Li et al., 2013); (2) vanilla neural model **DMCNN** (Chen et al., 2015); (3) the model incorporating syntactic knowledge, **dbRNN** (Sha et al., 2018); (4) state-of-the-art models on ED and EAE respectively, including **GatedGCN** (Lai et al., 2020) and **SemSyn-GTN** (Pouran Ben Veyseh et al., 2020); (5) a state-of-the-art EE model **RCEE_ER** (Liu et al., 2020), which tackle EE with machine reading comprehension (MRC) techniques. The last four models adopt PLMs to learn representations.

On MAVEN, we compare CLEVE with the official ED baselines set by Wang et al. (2020), including **DMCNN** (Chen et al., 2015), **BiLSTM** (Hochreiter and Schmidhuber, 1997), **BiLSTM+CRF**, **MOGANED** (Yan et al., 2019), **DMBERT** (Wang et al., 2019a), **BERT+CRF**.

Evaluation Results

The evaluation results are shown in Table 1 and Table 2. We can observe that: (1) CLEVE achieves significant improvements to its basic model RoBERTa on both ACE 2005 and MAVEN. The p-values under the t-test are 4×10^{-8}, 2×10^{-8} and 6×10^{-4} for ED on ACE 2005, EAE on ACE 2005, and ED on MAVEN, respectively. It also outperforms or achieves comparable results with

Metric (B-Cubed)	ED			EAE		
	P	R	F1	P	R	F1
LiberalEE	55.7	45.1	49.8	36.2	26.5	30.6
RoBERTa	44.3	24.9	31.9	24.2	17.3	20.2
RoBERTa+VGAE	47.0	26.8	34.1	25.6	17.9	21.1
CLEVE	**62.0**	**47.3**	**53.7**	**41.6**	**30.3**	**35.1**
w/o semantic	60.6	46.2	52.4	40.9	29.8	34.5
w/o structure	45.7	25.6	32.8	25.0	17.9	20.9
on ACE (AMR)	61.1	46.7	52.9	41.5	30.1	34.9

Table 3: Unsupervised "liberal" EE performance (%) of various models on ACE 2005.

Metric (B-Cubed)	ED		
	P	R	F1
RoBERTa	32.1	25.2	28.2
RoBERTa+VGAE	37.7	28.5	32.5
CLEVE	**55.6**	**46.4**	**50.6**
w/o semantic	53.2	44.8	48.6
w/o structure	32.8	26.1	29.1

Table 4: Unsupervised "liberal" EE performance (%) of various models on MAVEN.

all the baselines, including those using dependency parsing information (dbRNN, GatedGCN, SemSyn-GTN and MOGANED). This demonstrates the effectiveness of our proposed contrastive pre-training method and AMR semantic structure. It is noteworthy that RCEE_ER outperforms our method in EAE since its special advantages brought by reformulating EE as an MRC task to utilize sophisticated MRC methods and large annotated external MRC data. Considering that our method is essentially a pre-training method learning better event-oriented representations, CLEVE and RCEE_ER can naturally work together to improve EE further. (2) The ablation studies (comparisons between CLEVE and its w/o semantic or structure representations variants) indicate that both event semantic pre-training and event structure pre-training is essential to our method. (3) From the comparisons between CLEVE and its variants on ACE (golden) and ACE (AMR), we can see that the AMR parsing inevitably brings data noise compared to golden annotations, which results in a performance drop. However, this gap can be easily made up by the benefits of introducing large unsupervised data with pre-training.

4.4 Unsupervised "Liberal" EE

Dataset and Evaluation

In the unsupervised setting, we evaluate CLEVE on ACE 2005 and MAVEN with both objective automatic metrics and human evaluation. For the automatic evaluation, we adopt the extrinsic clustering evaluation metrics: B-Cubed Metrics (Bagga and Baldwin, 1998), including B-Cubed precision, recall and F1. The B-Cubed metrics evaluate the quality of cluster results by comparing them to golden standard annotations and have been shown to be effective (Amigó et al., 2009). For the human evaluation, we invite an expert to check the outputs of the models to evaluate whether the extracted events are complete and correctly clustered as well as whether all the events in text are discovered.

Baselines We compare CLEVE with reproduced **LiberalEE** (Huang et al., 2016), **RoBERTa** and **RoBERTa+VGAE**. **RoBERTa** here adopts the original RoBERTa (Liu et al., 2019) without event semantic pre-training to produce semantic representations for trigger and argument candidates in the same way as CLEVE, and encode the whole sentences to use the sentence embeddings (embeddings of the starting token <s>) as the needed event structure representations. **RoBERTa+VGAE** additionally adopts an unsupervised model Variational Graph Auto-Encoder (VGAE) (Kipf and Welling, 2016) to encode the AMR structures as event structure representations. RoBERTa+VGAE shares similar model architectures with CLEVE but is without our pre-training. Specially, for fair comparisons with LiberalEE, all the models in the unsupervised experiments adopt the same CAMR (Wang et al., 2015) as the AMR parser, including CLEVE pre-training. Moreover, we also study CLEVE variants as in the supervised setting. The **w/o semantic** variant replaces the CLEVE text encoder with a RoBERTa without event structure pre-training. The **w/o structure** variant only uses CLEVE text encoder in a similar way as **RoBERTa**. The **on ACE (AMR)** model is pre-trained with the parsed AMR structures of ACE test set. As shown in Huang et al. (2016), the AMR parsing is significantly superior to dependency parsing and frame semantic parsing on the unsupervised "liberal" event extraction task, hence we do not include baselines using other sentence structures in the experiments.

Evaluation Results

The automatic evaluation results are shown in Table 3 and Table 4. As the human evaluation is laborious and expensive, we only do human

Metric (Human)	ED			EAE		
	P	R	F1	P	R	F1
LiberalEE	51.2	46.9	49.0	33.5	27.2	30.0
CLEVE	**60.4**	**48.4**	**53.7**	**39.4**	**31.1**	**34.8**

Table 5: Unsupervised "liberal" EE human-evaluation performance (%) on ACE 2005.
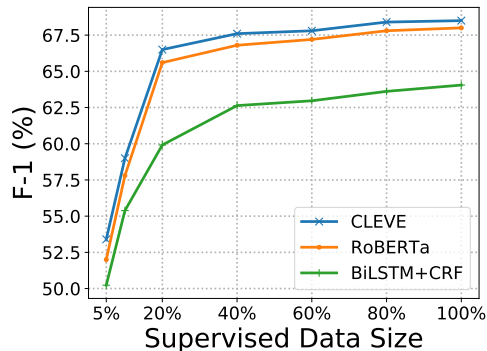
Figure 3: Supervised ED performance (F-1) on MAVEN with different training data size.

	AMR 1.0	ACE 2005		MAVEN
	Parsing	ED	EAE	ED
Wang et al. (2015)	62.0	79.8	61.1	68.5
Xu et al. (2020)	79.1	80.6	61.5	69.0

Table 6: Supervised results (F1,%) on ACE 2005 and MAVEN of CLEVE using different AMR parsers, as well as the performance (F1,%) of the parsers on AMR 1.0 (LDC2015E86) dataset.

	ACE 2005		MAVEN
	ED	EAE	ED
NYT	**79.8**	**61.1**	68.5
w/o semantic	77.4	59.7	68.2
w/o structure	79.5	60.7	68.4
Wikipedia	79.1	60.4	**68.8**
w/o semantic	77.3	59.5	68.4
w/o structure	78.8	60.0	68.6

Table 7: Supervised results (F1,%) on ACE 2005 and MAVEN of CLEVE pre-trained on different corpora.

evaluations for CLEVE and the most competitive baseline LiberalEE on ACE 2005, and the results are shown in Table 5. We can observe that: (1) CLEVE significantly outperforms all the baselines, which shows its superiority in both extracting event instances and discovering event schemata. (2) RoBERTa ignores the structure information. Although RoBERTa+VAGE encodes event structures with VGAE, the semantic representations of RoBERTa and the structure representations of VGAE are distinct and thus cannot work together well. Hence the two models even underperform LiberalEE, while the two representations of CLEVE can collaborate well to improve "liberal" EE. (3) In the ablation studies, the discarding of event structure pre-training results in a much more significant performance drop than in the supervised setting, which indicates event structures are essential to discovering new event schemata.

5 Analysis

5.1 Effect of Supervised Data Size

In this section, we study how the benefits of pre-training change along with the available supervised data size. We compare the ED performance on MAVEN of CLEVE, RoBERTa and a non-pre-training model BiLSTM+CRF when trained on different proportions of randomly-sampled MAVEN training data in Figure 3. We can see that the im-

provements of CLEVE compared to RoBERTa and the pre-training models compared to the non-pre-training model are generally larger when less supervised data available. It indicates that CLEVE is especially helpful for low-resource EE tasks, which is common since the expensive event annotation.

5.2 Effect of AMR Parsers

CLEVE relies on automatic AMR parsers to build self-supervision signals for large unsupervised data. Intuitively, the performance of AMR parsers will influence CLEVE performance. To analyze the effect of different AMR parsing performance, we compare supervised EE results of CLEVE models using the established CAMR (Wang et al., 2016) and a new state-of-the-art parser (Xu et al., 2020) during pre-training in Table 6. We can see that a better AMR parser intuitively brings better EE performance, but the improvements are not so significant as the corresponding AMR performance improvement, which indicates that CLEVE is generally robust to the errors in AMR parsing.

5.3 Effect of Pre-training Domain

Pre-training on similar text domains may further improve performance on corresponding downstream tasks (Gururangan et al., 2020; Gu et al., 2020). To analyze this effect, we evaluate the supervised EE performance of CLEVE pre-trained on NYT and English Wikipedia in Table 7. We can see pre-training on a similar domain (NYT for

ACE 2005, Wikipedia for MAVEN) surely benefits CLEVE on corresponding datasets. On ACE 2005, although Wikipedia is 2.28 times as large as NYT, CLEVE pre-trained on it underperforms CLEVE pre-trained on NYT (both in the news domain). Moreover, we can see the in-domain benefits mainly come from the event semantics rather than structures in CLEVE framework (from the comparisons between the **w/o semantic** and **w/o structure** results). It suggests that we can develop domain adaptation techniques focusing on semantics for CLEVE, and we leave it to future work.

6 Conclusion and Future work

In this paper, we propose CLEVE, a contrastive pre-training framework for event extraction to utilize the rich event knowledge lying in large unsupervised data. Experiments on two real-world datasets show that CLEVE can achieve significant improvements in both supervised and unsupervised "liberal" settings. In the future, we will (1) explore other kinds of semantic structures like the frame semantics and (2) attempt to overcome the noise in unsupervised data brought by the semantic parsers.

Acknowledgement

Ethical Considerations

We discuss the ethical considerations and broader impact of the proposed CLEVE method in this section: (1) **Intellectual property**. NYT and ACE 2005 datasets are obtained from the linguistic data consortium (LDC), and are both licensed to be used for research. MAVEN is publicly shared under the CC BY-SA 4.0 license[3]. The Wikipedia corpus is obtained from the Wikimedia dump[4], which is shared under the CC BY-SA 3.0 license[5]. The invited expert is fairly paid according to agreed working hours. (2) **Intended use**. CLEVE improves event extraction in both supervised and unsupervised settings, i.e., better extract structural events from diverse raw text. The extracted events then help people to get information conveniently and can be used to build a wide range of application systems like information retrieval (Glavaš and Šnajder, 2014) and knowledge base population (Ji and Grishman, 2011). As extracting events is fundamental to various applications, the failure cases and potential bias in EE methods also have a significant negative impact. We encourage the community to put more effort into analyzing and mitigating the bias in EE systems. Considering CLEVE does not model people's characteristics, we believe CLEVE will not bring significant additional bias. (3) **Misuse risk.** Although all the datasets used in this paper are public and licensed, there is a risk to use CLEVE methods on private data without authorization for interests. We encourage the regulators to make efforts to mitigate this risk. (4) **Energy and carbon costs.** To estimate the energy and carbon costs, we present the computing platform and running time of our experiments in Appendix E for reference. We will also release the pre-trained checkpoints to avoid the additional carbon costs of potential users. We encourage the users to try model compression techniques like distillation and quantization in deployment to reduce carbon costs.

References

Jacqueline Aguilar, Charley Beller, Paul McNamee, Benjamin Van Durme, Stephanie Strassel, Zhiyi Song, and Joe Ellis. 2014. A comparison of the events and relations across ACE, ERE, TAC-KBP, and FrameNet annotation standards. In *Proceedings of the Second Workshop on EVENTS: Definition, Detection, Coreference, and Representation*, pages 45–53.

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

Enrique Amigó, Julio Gonzalo, Javier Artiles, and Felisa Verdejo. 2009. A Comparison of Extrinsic Clustering Evaluation Metrics Based on Formal Constraints. *Inf. Retr.*, 12(4):461–486.

Amit Bagga and Breck Baldwin. 1998. Entity-Based Cross-Document Coreferencing Using the Vector

---

[3] https://creativecommons.org/licenses/by-sa/4.0/
[4] https://dumps.wikimedia.org/

[5] https://creativecommons.org/licenses/by-sa/3.0/

Space Model. In *Proceedings of ACL-COLING*, pages 79–85.

Livio Baldini Soares, Nicholas FitzGerald, Jeffrey Ling, and Tom Kwiatkowski. 2019. Matching the Blanks: Distributional Similarity for Relation Learning. In *Proceedings of ACL*, pages 2895–2905.

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2013. Abstract Meaning Representation for Sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 178–186.

Nathanael Chambers. 2013. Event Schema Induction with a Probabilistic Entity-Driven Model. In *Proceedings of EMNLP*, pages 1797–1807.

Nathanael Chambers and Dan Jurafsky. 2011. Template-Based Information Extraction without the Templates. In *Proceedings of ACL-HLT*, pages 976–986.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A Simple Framework for Contrastive Learning of Visual Representations. In *Proceedings of ICML*, pages 1597–1607.

Yubo Chen, Shulin Liu, Xiang Zhang, Kang Liu, and Jun Zhao. 2017. Automatically Labeled Data Generation for Large Scale Event Extraction. In *Proceedings of ACL*, pages 409–419.

Yubo Chen, Liheng Xu, Kang Liu, Daojian Zeng, and Jun Zhao. 2015. Event extraction via dynamic multi-pooling convolutional neural networks. In *Proceedings of ACL-IJCNLP*, pages 167–176.

Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. 2020. ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators. In *Proceedings of ICLR*.

Wanyun Cui, Guangyu Zheng, and Wei Wang. 2020. Unsupervised Natural Language Inference via Decoupled Multimodal Contrastive Learning. In *Proceedings of EMNLP*, pages 5511–5520.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186.

Joe Ellis, Jeremy Getman, Dana Fore, Neil Kuster, Zhiyi Song, Ann Bies, and Stephanie M Strassel. 2015. Overview of linguistic resources for the TAC KBP 2015 evaluations: Methodologies and results. In *TAC*.

Joe Ellis, Jeremy Getman, Dana Fore, Neil Kuster, Zhiyi Song, Ann Bies, and Stephanie M Strassel. 2016. Overview of Linguistic Resources for the TAC KBP 2016 Evaluations: Methodologies and Results. In *TAC*.

Jeremy Getman, Joe Ellis, Zhiyi Song, Jennifer Tracey, and Stephanie Strassel. 2017. Overview of linguistic resources for the tac kbp 2017 evaluations: Methodologies and results. In *TAC*.

Goran Glavaš and Jan Šnajder. 2014. Event graphs for information retrieval and multi-document summarization. *Expert systems with applications*, 41(15):6904–6916.

Yuxian Gu, Zhengyan Zhang, Xiaozhi Wang, Zhiyuan Liu, and Maosong Sun. 2020. Train No Evil: Selective Masking for Task-Guided Pre-Training. In *Proceedings of EMNLP*, pages 6966–6974.

Prashant Gupta and Heng Ji. 2009. Predicting Unknown Time Arguments based on Cross-Event Propagation. In *Proceedings of ACL-IJCNLP*, pages 369–372.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't Stop Pretraining: Adapt Language Models to Domains and Tasks. In *Proceedings of ACL*, pages 8342–8360.

Raia Hadsell, Sumit Chopra, and Yann LeCun. 2006. Dimensionality Reduction by Learning an Invariant Mapping. In *Proceedings of CVPR*, volume 2, pages 1735–1742.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. 2020. Momentum Contrast for Unsupervised Visual Representation Learning. In *Proceedings of CVPR*, pages 9726–9735.

R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio. 2019. Learning deep representations by mutual information estimation and maximization. In *Proceedings of ICLR*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Lifu Huang, Taylor Cassidy, Xiaocheng Feng, Heng Ji, Clare R. Voss, Jiawei Han, and Avirup Sil. 2016. Liberal Event Extraction and Event Schema Induction. In *Proceedings of ACL*, pages 258–268.

Lifu Huang and Heng Ji. 2020. Semi-supervised New Event Type Induction and Event Detection. In *Proceedings of EMNLP*, pages 718–724.

Lifu Huang, Heng Ji, Kyunghyun Cho, Ido Dagan, Sebastian Riedel, and Clare Voss. 2018. Zero-Shot Transfer Learning for Event Extraction. In *Proceedings of ACL*, pages 2160–2170.

Dan Iter, Kelvin Guu, Larry Lansing, and Dan Jurafsky. 2020. Pretraining with Contrastive Sentence Objectives Improves Discourse Performance of Language Models. In *Proceedings of ACL*, pages 4859–4870.

Heng Ji and Ralph Grishman. 2008. Refining event extraction through cross-document inference. In *Proceedings of ACL*, pages 254–262.

Heng Ji and Ralph Grishman. 2011. Knowledge Base Population: Successful Approaches and Challenges. In *Proceedings of ACL*, pages 1148–1158.

Thomas N Kipf and Max Welling. 2016. Variational graph auto-encoders. *NIPS Workshop on Bayesian Deep Learning*.

Lingpeng Kong, Cyprien de Masson d'Autume, Wang Ling, Lei Yu, Zihang Dai, and Dani Yogatama. 2020. A Mutual Information Maximization Perspective of Language Representation Learning. In *Proceedings of ICLR*.

Viet Dac Lai, Tuan Ngo Nguyen, and Thien Huu Nguyen. 2020. Event Detection: Gate Diversity and Syntactic Importance Scores for Graph Convolution Neural Networks. In *Proceedings of EMNLP*, pages 5405–5411.

Qi Li, Heng Ji, and Liang Huang. 2013. Joint event extraction via structured prediction with global features. In *Proceedings of ACL*, pages 73–82.

Shasha Liao and Ralph Grishman. 2010. Using document level cross-event inference to improve event extraction. In *Proceedings of ACL*, pages 789–797.

Jian Liu, Yubo Chen, Kang Liu, Wei Bi, and Xiaojiang Liu. 2020. Event Extraction as Machine Reading Comprehension. In *Proceedings of EMNLP*, pages 1641–1651.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *CoRR*, abs/1907.11692.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. In *Proceedings of ICLR*.

Kiem-Hieu Nguyen, Xavier Tannier, Olivier Ferret, and Romaric Besançon. 2015. Generative Event Schema Induction with Entity Disambiguation. In *Proceedings of ACL*, pages 188–197.

Thien Nguyen and Ralph Grishman. 2018. Graph convolutional networks with argument-aware pooling for event detection. In *Proceedings of AAAI*, pages 5900–5907.

Thien Huu Nguyen, Kyunghyun Cho, and Ralph Grishman. 2016. Joint event extraction via recurrent neural networks. In *Proceedings of NAACL*, pages 300–309.

Thien Huu Nguyen and Ralph Grishman. 2015. Event Detection and Domain Adaptation with Convolutional Neural Networks. In *Proceedings of ACL*, pages 365–371.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. In *Proceedings of NIPS*.

Hao Peng, Tianyu Gao, Xu Han, Yankai Lin, Peng Li, Zhiyuan Liu, Maosong Sun, and Jie Zhou. 2020. Learning from Context or Names? An Empirical Study on Neural Relation Extraction. In *Proceedings of EMNLP*, pages 3661–3672.

Amir Pouran Ben Veyseh, Tuan Ngo Nguyen, and Thien Huu Nguyen. 2020. Graph Transformer Networks with Syntactic and Semantic Structures for Event Argument Extraction. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3651–3661.

Jiezhong Qiu, Qibin Chen, Yuxiao Dong, Jing Zhang, Hongxia Yang, Ming Ding, Kuansan Wang, and Jie Tang. 2020. GCC: Graph Contrastive Coding for Graph Neural Network Pre-Training. In *Proceedings of KDD*, page 1150–1160.

Evan Sandhaus. 2008. The new york times annotated corpus. *Linguistic Data Consortium*, 6(12):e26752.

Lei Sha, Sujian Li, Baobao Chang, and Zhifang Sui. 2016. Joint Learning Templates and Slots for Event Schema Induction. In *Proceedings of NAACL-HLT*, pages 428–434.

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 *Proceedings of AAAI*, pages 5916–5923.

Meihan Tong, Bin Xu, Shuai Wang, Yixin Cao, Lei Hou, Juanzi Li, and Jun Xie. 2020. Improving event detection via open-domain trigger knowledge. In *Proceedings of ACL*, pages 5887–5897.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Proceedings of NIPS*, pages 5998–6008.

Ulrike Von Luxburg. 2007. A tutorial on spectral clustering. *Statistics and computing*, 17(4):395–416.

David Wadden, Ulme Wennberg, Yi Luan, and Hannaneh Hajishirzi. 2019. Entity, Relation, and Event Extraction with Contextualized Span Representations. In *Proceedings of EMNLP-IJCNLP*, pages 5784–5789.

Christopher Walker, Stephanie Strassel, Julie Medero, and Kazuaki Maeda. 2006. ACE 2005 multilingual training corpus. *Linguistic Data Consortium*, 57.

Chuan Wang, Sameer Pradhan, Xiaoman Pan, Heng Ji, and Nianwen Xue. 2016. CAMR at SemEval-2016 Task 8: An Extended Transition-based AMR Parser. In *Proceedings of SemEval*, pages 1173–1178.

Chuan Wang, Nianwen Xue, and Sameer Pradhan. 2015. Boosting Transition-based AMR Parsing with Refined Actions and Auxiliary Analyzers. In *Proceedings of ACL-IJCNLP*, pages 857–862.

Xiaozhi Wang, Xu Han, Zhiyuan Liu, Maosong Sun, and Peng Li. 2019a. Adversarial Training for Weakly Supervised Event Detection. In *Proceedings of NAACL-HLT*, pages 998–1008.

Xiaozhi Wang, Ziqi Wang, Xu Han, Wangyi Jiang, Rong Han, Zhiyuan Liu, Juanzi Li, Peng Li, Yankai Lin, and Jie Zhou. 2020. MAVEN: A Massive General Domain Event Detection Dataset. In *Proceedings of EMNLP*, pages 1652–1671.

Xiaozhi Wang, Ziqi Wang, Xu Han, Zhiyuan Liu, Juanzi Li, Peng Li, Maosong Sun, Jie Zhou, and Xiang Ren. 2019b. HMEAE: Hierarchical Modular Event Argument Extraction. In *Proceedings of EMNLP-IJCNLP*, pages 5777–5783.

Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. 2018. Unsupervised Feature Learning via Nonparametric Instance Discrimination. In *Proceedings of CVPR*, pages 3733–3742.

Dongqin Xu, Junhui Li, Muhua Zhu, Min Zhang, and Guodong Zhou. 2020. Improving AMR Parsing with Sequence-to-Sequence Pre-training. In *Proceedings of EMNLP*, pages 2501–2511.

Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. 2019. How Powerful are Graph Neural Networks? In *Proceedings of ICLR*.

Haoran Yan, Xiaolong Jin, Xiangbin Meng, Jiafeng Guo, and Xueqi Cheng. 2019. Event Detection with Multi-Order Graph Convolution and Aggregated Attention. In *Proceedings of EMNLP-IJCNLP*, pages 5766–5770.

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 ACL*, pages 5284–5294.

Yi-Ting Yeh and Yun-Nung Chen. 2019. QAInfomax: Learning Robust Question Answering System by Mutual Information Maximization. In *Proceedings of EMNLP-IJCNLP*, pages 3370–3375.

Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. 2020. Graph contrastive learning with augmentations. In *Proceedings of NeurIPS*, pages 5812–5823.

Yanqiao Zhu, Yichen Xu, Feng Yu, Qiang Liu, Shu Wu, and Liang Wang. 2020. Deep Graph Contrastive Representation Learning. In *ICML Workshop on Graph Representation Learning and Beyond*.

A Downstream Adaptation of CLEVE

In this section, we introduce how to adapt pre-trained CLEVE to make the event semantic and structure representations work together in downstream event extraction settings in detail, including supervised EE and unsupervised "liberal" EE.

A.1 Supervised EE

In supervised EE, we fine-tune the pre-trained text encoder and graph encoder of CLEVE with annotated data. We formulate both event detection (ED) and event argument extraction (EAE) as multi-class classification tasks. An instance is defined as a sentence with a trigger candidate for ED, and a sentence with a given trigger and an argument candidate for EAE. The key question here is how to obtain features of an instance to be classified.

For the event semantic representation, we adopt dynamic multi-pooling to aggregate the embeddings produced by text encoder into a unified semantic representation $\mathbf{x}_{sem}$ following previous work (Chen et al., 2015; Wang et al., 2019a,b). Moreover, we also insert special markers to indicate candidates as in pre-training (Section 3.2). For the event structure representation, we parse the sentence into an AMR graph and find the corresponding node v of the trigger/argument candidate to be classified. Following Qiu et al. (2020), we encode v and its one-hop neighbors with the graph encoder to get the desired structure representation $\mathbf{g}_{str}$. The initial node representation is also obtained with the text encoder as introduced in Section 3.3.

We concatenate $\mathbf{x}_{sem}$ and $\mathbf{g}_{str}$ as the instance embedding and adopt a multi-layer perceptron along with softmax to get the logits. Then we fine-tune CLEVE with cross-entropy loss.

A.2 Unsupervised "Liberal" EE

Unsupervised "liberal" EE requires to discover event instances and event schemata only from raw text. We follow the pipeline of Huang et al. (2016) to parse sentences into AMR graphs and identify trigger and argument candidates with the AMR structures. We also cluster the candidates to get event instances and schemata with the joint constraint clustering algorithm (Huang et al., 2016), which requires semantic representations of the trigger and argument candidates as well as the event structure representations. The details of this clustering algorithm is introduced in Appendix B. Here we straightforwardly use the corresponding text

span representations (Section 3.2) as semantic representations and encode the whole AMR graphs with the graph encoder to get desired event structure representations.

B   Joint Constraint Clustering Algorithm

In the unsupervised "liberal" event extraction (Huang et al., 2016), the joint constraint clustering algorithm is introduced to get trigger and argument clusters given trigger and argument candidate representations. CLEVE focuses on learning event-specific representations and can use any clustering algorithm. To fairly compare with Huang et al. (2016), we also use the joint constraint clustering algorithm in our unsupervised evaluation. Hence we briefly introduce this algorithm here.

B.1   Preliminaries

The input of this algorithm contains a trigger candidate set T and an argument candidate set A as well as their semantic representations E_g^T and E_g^A, respectively. There is also an event structure representation E_R^t for each trigger t. We also previously set the ranges of the numbers of resulting trigger and argument clusters: the minimal and maximal number of trigger clusters $K_T^{\min}$, $K_T^{\max}$ as well as the minimal and maximal number of argument clusters $K_A^{\min}$, $K_A^{\max}$. The algorithm will output the optimal trigger clusters $C^T = \{C_1^T, ..., C_{K_T}^T\}$ and argument clusters $C^A = \{C_1^A, ..., C_{K_A}^A\}$.

B.2   Similarity Functions

The clustering algorithm requires to define trigger-trigger similarities and argument-argument similarities. Huang et al. (2016) first defines the constraint function f:

$$f(\mathcal{P}_1, \mathcal{P}_2) = \log(1 + \frac{|\mathcal{L}_1 \cap \mathcal{L}_2|}{|\mathcal{L}_1 \cup \mathcal{L}_2|}). \qquad (4)$$

When $\mathcal{P}_1$ and $\mathcal{P}_2$ are two triggers, $\mathcal{L}_i$ has tuple elements $(\mathcal{P}_i, r, \mathrm{id}(a))$, which means the argument a has a relation r to trigger $\mathcal{P}_i$. $\mathrm{id}(a)$ is the cluster ID for the argument a. When $\mathcal{P}_i$ is arguments, $\mathcal{L}_i$ changes to corresponding triggers and semantic relations accordingly.

Hence the similarity functions are defined as:

$$\mathrm{sim}(t_1, t_2) = \lambda \, \mathrm{sim}_{\cos}(E_g^{t_1}, E_g^{t_2}) + f(t_1, t_2)$$
$$+ (1 - \lambda) \frac{\sum_{r \in R_{t_1} \cap R_{t_2}} \mathrm{sim}_{\cos}(E_r^{t_1}, E_r^{t_2})}{|R_{t_1} \cap R_{t_2}|},$$

$$\mathrm{sim}(a_1, a_2) = \mathrm{sim}_{\cos}(E_g^{a_1}, E_g^{a_2}) + f(a_1, a_2)$$

$$(5)$$

where E_g^t and E_g^a are trigger and argument semantic representations, respectively. R_t is the AMR relation set in the parsed AMR graph of trigger t. E_r^t denotes the event structure representation of the node that has a semantic relation r to trigger t in the event structure. λ is a hyper-parameter. $\mathrm{sim}_{\cos}(\cdot, \cdot)$ is the cosine similarity.

B.3   Objective

Huang et al. (2016) also defines an objective function $O(\cdot, \cdot)$ to evaluate the quality of trigger clusters $C^T = \{C_1^T, ..., C_{K_T}^T\}$ and argument clusters $C^A = \{C_1^A, ..., C_{K_A}^A\}$. It is defined as follows:

$$O(C^T, C^A) = D_{\mathrm{inter}}(C^T) + D_{\mathrm{intra}}(C^T)$$
$$+ D_{\mathrm{inter}}(C^A) + D_{\mathrm{intra}}(C^A),$$

$$D_{\mathrm{inter}}(C^{\mathcal{P}}) = \sum_{i \neq j=1}^{K_{\mathcal{P}}} \sum_{u \in C_i^{\mathcal{P}}, v \in C_j^{\mathcal{P}}} \mathrm{sim}(\mathcal{P}_u, \mathcal{P}_v), \quad (6)$$

$$D_{\mathrm{intra}}(C^{\mathcal{P}}) = \sum_{i=1}^{K_{\mathcal{P}}} \sum_{u,v \in C_i^{\mathcal{P}}} (1 - \mathrm{sim}(\mathcal{P}_u, \mathcal{P}_v)),$$

where $D_{\mathrm{inter}}(\cdot)$ measures the agreement across clusters, and $D_{\mathrm{intra}}(\cdot)$ measures the disagreement within clusters. The clustering algorithm iteratively minimizes the objective function.

B.4   Overall Pipeline

This algorithm updates its clustering results iteratively. At first, it uses the Spectral Clustering algorithm (Von Luxburg, 2007) to get initial clustering results. Then for each iteration, it updates clustering results and the best objective value using previous clustering results. It selects the clusters with the minimum O value as the final result. The overall pipeline is shown in Algorithm 1.

C   Subgraph Sampling

In the AMR subgraph discrimination task of event structure pre-training, we need to sample subgraphs from the parsed AMR graphs for contrastive pre-training. Here we adopt the subgraph sampling strategy introduced by Qiu et al. (2020), which consists of the random walk with restart (RWR), subgraph induction and anonymization:

- **Random walk with restart** first randomly chooses a starting node (the *ego*) from the AMR graph to be sampled from. The *ego* must be a root node, i.e., there is no directed edge in the AMR graph pointing to the node. Then we treat the AMR graph as an undirected graph

Algorithm 1 Joint Constraint Clustering Algorithm

Input: Trigger candidate set T, Argument candidate set A, their semantic representations E_g^T and E_g^A, structure representations E_R^t for each trigger t, the minimal and maximal number of trigger clusters $K_T^{\min}$, $K_T^{\max}$ as well as the minimal and maximal number of argument clusters $K_A^{\min}$, $K_A^{\max}$;

Output: Optimal trigger clusters $C^T = \{C_1^T, ..., C_{K_T}^T\}$ and argument clusters $C^A = \{C_1^A, ..., C_{K_A}^A\}$;

- $O_{\min} = \infty, C^T = \emptyset, C^A = \emptyset$

- For $K_T = K_T^{\min}$ to $K_T^{\max}$, $K_A = K_A^{\min}$ to $K_A^{\max}$
    - Clustering with Spectral Clustering Algorithm:
    - $C_{\text{curr}}^T = \text{spectral}(T, E_g^T, E_R^T, K_T, C_{\text{curr}}^A)$
    - $C_{\text{curr}}^A = \text{spectral}(A, E_g^A, K_A)$
    - $O_{\text{curr}} = O(C_{\text{curr}}^T, C_{\text{curr}}^A)$
    - if $O_{\text{curr}} \leq O_{\min}$
        * $O_{\min} = O_{\text{curr}}, C^T = C_{\text{curr}}^T, C^A = C_{\text{curr}}^A$
    - while iterate time ≤ 10
        * $C_{\text{curr}}^T = \text{spectral}(T, E_g^T, E_R^T, K_T, C_{\text{curr}}^A)$
        * $C_{\text{curr}}^A = \text{spectral}(A, E_g^A, K_A, C_{\text{curr}}^T)$
        * $O_{\text{curr}} = O(C_{\text{curr}}^T, C_{\text{curr}}^A)$
        * if $O_{\text{curr}} \leq O_{\min}$
            · $O_{\min} = O_{\text{curr}}, C^T = C_{\text{curr}}^T, C^A = C_{\text{curr}}^A$

- return $O_{\min}, C^T, C^A$

Batch size	40
Learning rate	1×10^{-5}
Adam ϵ	1×10^{-8}
Adam β_1	0.9
Adam β_2	0.999
Trigger negative sampling size m_t	9
Argument negative sampling size m_a	30
Max sequence length	128
#parameters of text encoder	355M

Table 8: Hyperparameters for the event semantic pre-training.

Batch size	1024
Restart probability	0.8
Temperature	0.07
Warmup steps	7,500
Weight decay	1×10^{-5}
Training steps	75,000
Learning rate	0.005
Adam ϵ	1×10^{-8}
Adam β_1	0.9
Adam β_2	0.999
Number of layers	5
Dropout rate	0.5
Hidden dimensions	64
#parameters of graph encoder	0.2M

Table 9: Hyperparameters for the event structure pre-training.

and do random walks starting from the *ego*. At each step, the random walk with a probability to return to the *ego* and restart. When all the neighbouring nodes of the current node have been visited, the RWR ends.

- **Subgraph induction** is to take the induced subgraph of the node set obtained with RWR as the sampled subgraphs.

- **Anonymization** is to randomly shuffle the indices of the nodes in the sampled subgraph to avoid overfitting to the node representations.

In our event structure pre-training, we take subgraphs of the same sentence (AMR graph) as positive pairs. But, ideally, the two subgraphs in a positive pair should be taken from the same event rather than only the same sentence. However, it is hard to unsupervisedly determine which parts of an AMR graph belong to the same event. We think this task is almost as hard as event extraction itself. The rule used in the event semantic pre-training only handles the `ARG`, `time` and `location` relations, and for the other about 100 AMR relations, we cannot find an effective method to determine

which event their edges belong to. Hence, to take advantage of all the structure information, we adopt a simple assumption that the subgraphs from the same sentence express the same event (or at least close events) to design the subgraph sampling part here. We will explore more sophisticated subgraph-sampling strategies in our future work.

D Hyperparameter Setup

D.1 Pre-training Hyperparameters

During pre-training, we manually tune the hyperparameters and select the models by the losses on a held-out validation set with $1,000$ sentences. The event structure pre-training hyperparameters mainly follow the E2E model of Qiu et al. (2020). Table 8 and Table 9 show the best-performing hyper-parameters used in experiments of the event semantic pre-training and event structure pre-training, respectively.

		40
Batch size		40
Training epoch		30
Learning rate		1×10^{-5}
Adam ϵ		1×10^{-8}
Adam β_1		0.9
Adam β_2		0.999
Max sequence length		128

Table 10: Fine-tuning hyperparameters for CLEVE and RoBERTa in the supervised setting.

D.2 Fine-tuning Hyperparameters

CLEVE in the unsupervised "liberal" setting directly uses the pre-trained representations and hence does not have additional hyperparameters. For the fine-tuning in the supervised setting, we manually tune the hyperparameters by 10 trials. In each trial, we train the models for 30 epochs and select models by their F1 scores on the validation set. Table 10 shows the best fine-tuning hyperparameters for CLEVE models and RoBERTa. For the other baselines, we take their reported results.

E Training Details

For reproducibility and estimating energy and carbon costs, we report the computing infrastructures and average runtime of experiments as well as validation performance.

E.1 Pre-training Details

For pre-training, we use 8 RTX 2080 Ti cards. The event semantic pre-training takes 12.3 hours. The event structure pre-training takes 60.2 hours.

E.2 Fine-tuning/Inference Details

During the fine-tuning in the supervised setting and the inference in the unsupervised "liberal" setting, we also use 8 RTX 2080 Ti cards.

For the supervised EE experiments, Table 11 and Table 12 show the runtime and the results on the validation set of the model implemented by us.

In the unsupervised "liberal" setting, we only do inference and do not involve the validation. We report the runtime of our models in Table 13.

	ED			EAE			Runtime
Metric	P	R	F1	P	R	F1	mins
RoBERTa	72.9	75.2	74.0	54.3	62.6	58.2	344
CLEVE	73.7	79.4	76.4	56.2	66.0	60.7	410
w/o semantic	72.1	77.9	74.9	54.5	65.6	59.5	422
w/o structure	73.2	80.2	76.5	56.3	65.4	60.5	355
on ACE (golden)	71.0	77.1	73.9	55.0	65.8	59.9	401
on ACE (AMR)	70.2	77.3	73.6	54.1	65.5	59.3	408

Table 11: Supervised EE performance (%) of various models on ACE 2005 validation set and the models' average fine-tuning runtime.

	ED			Runtime
Metric	P	R	F1	mins
RoBERTa	65.3	71.4	68.2	530
CLEVE	66.1	70.2	68.1	572
w/o semantic	66.5	69.3	67.9	588
w/o structure	65.4	71.7	68.4	549

Table 12: Supervised EE performance (%) of various models on MAVEN validation set and the models' average fine-tuning runtime.

	ACE 2005	MAVEN
RoBERTa	12 mins	29 mins
RoBERTa+VGAE	17 mins	36 mins
CLEVE	15 mins	33 mins
w/o semantic	15 mins	32 mins
w/o structure	14 mins	26 mins

Table 13: Average runtime of various models on ACE and MAVEN for the unsupervised "liberal" EE.