Soft Syntactic Reinforcement for Neural Event Extraction

Anran Hao[1,2], **Jian Su**[2], **Shuo Sun**[2], **Teo Yong Sen**[1]
[1]Nanyang Technological University, Singapore
[2]Institute for Infocomm Research (I^{2}R), A*STAR, Singapore
[1]`{S190003, teoy0050}@ntu.edu.sg`
[2]`{Sun_Shuo, sujian}@i2r.a-star.edu.sg`

Abstract

Recent event extraction (EE) methods rely on pre-trained language models (PLMs) but still suffer from errors due to a lack of syntactic knowledge. While syntactic information is crucial for EE, there is a need for effective methods to incorporate syntactic knowledge into PLMs. To address this gap, we present a novel method to incorporate syntactic information into PLM-based models for EE, which do not require external syntactic parsers to produce syntactic features of task data. Instead, our proposed soft syntactic reinforcement (SSR) mechanism learns to select syntax-related dimensions of PLM representation during pretraining on a standard dependency corpus. The adapted PLM weights and the syntax-aware representation then facilitate the model's prediction over the task data. On both sentence-level and document-level EE benchmark datasets, our proposed method achieves state-of-the-art results, outperforming baseline models and existing syntactic reinforcement methods. To the best of our knowledge, this is the first work in this direction. Our code is available at `https://github.com/Anran971/sre-naacl25`.

1 Introduction

Event Extraction (EE), which obtains poly-element structured information regarding *what happens* from unstructured text, is a challenging task in natural language processing. Among EE literature, syntax information has been found beneficial for the task (Nguyen et al., 2016; Yan et al., 2019; Liu et al., 2019a; Lai et al., 2020). Syntactic features such as dependency trees provide clues for the models to better learn the interrelations between candidate trigger words and respective entities in sentences.

With the advent of large pre-trained language models (PLMs), there has been a notable shift towards neural EE methods (Xiang and Wang, 2019),

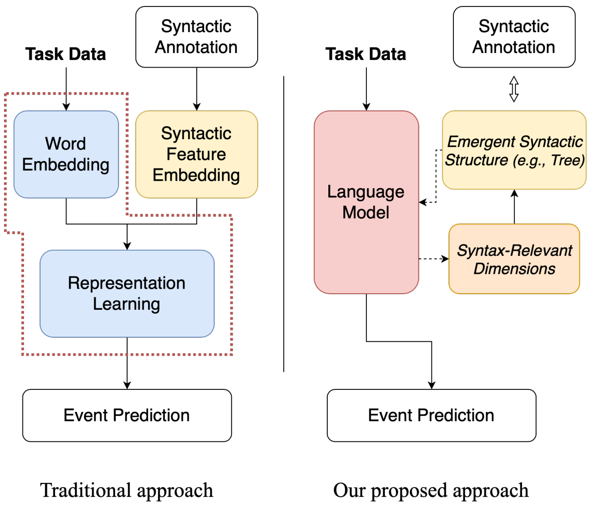

Figure 1: Illustration of the traditional approach vs our proposed approach to integrating syntactic knowledge for event extraction.

which have demonstrated impressive performance gains. As shown in Figure 1, prior to PLM, the traditional approach to EE relies on a "word embedding + representation learning" paradigm (red dotted lines), in which syntactic annotation is added to enhance the system via syntactic feature embedding. In other words, separate embeddings for words and syntactic features are trained and then combined for task-specific representation learning and prediction. The latest EE studies based on PLM replace the "word embedding + representation learning" structure with a powerful language model (the red block). With sufficient self-supervised pre-training, these PLM-based models significantly improve task-specific representation learning and lead to new state-of-the-art results in EE.

However, qualitative analyses reveal that PLM-based EE models still make certain errors due to a deficiency in syntactic structure knowledge (Lin et al., 2020; Du and Cardie, 2020b) for event extraction. We empirically found that the conventional

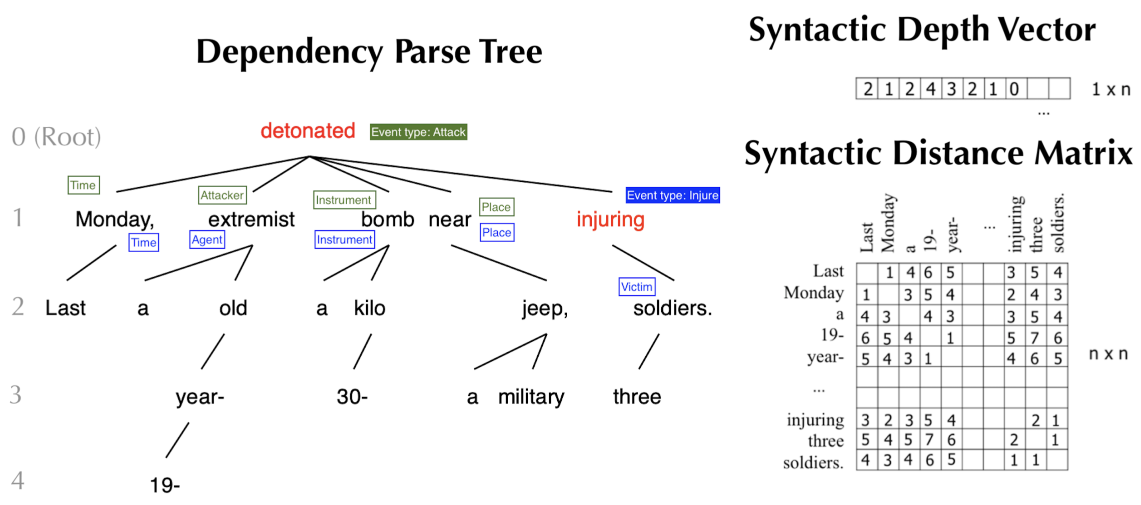

Figure 2: Our method relies on dependency parse trees, a classical structure to represent syntax. Take the sentence "*Last Monday, a 19-year-old extremist detonated a 30-kilo bomb near a military jeep, injuring three soldiers.*" as an example, its dependency parse tree can be converted to a syntactic depth vector or a syntactic distance matrix.

way to inject syntactic knowledge into EE models, i.e., to use an external syntax parser to generate a syntactic representation on task-specific data, and then combine them with the PLM encoding, is ineffective. Unless human-annotated parses on task data are used for training, the external representations are largely redundant and do not lead to performance gain for downstream NLP tasks. This aligns with probing studies on PLM (Ruder et al., 2019; Hewitt and Manning, 2019; Maudslay et al., 2020; Li et al., 2020; Newman et al., 2021), which found that syntax trees can be implicitly embedded in PLM's vector geometry.

In this work, we introduce a novel approach to syntactically enhance PLM-based EE models, using a Soft Syntactic Reinforcement (SSR) mechanism. Without the need for syntactically annotating task data, our approach leverages a general-purpose syntactic resource, the Penn Tree Bank (PTB) dependency dataset, for reinforcing implicit syntactic knowledge learned by PLMs. The SSR mechanism learns transformation matrices to activate syntax-relevant dimensions for emergent syntactic structures in a PLM and then uses the enhanced syntactic representations for event prediction. To the best of our knowledge, this is the first work in this direction. Our results suggest that syntactic reinforcement mechanisms are important for event extraction research and contribute to the understanding of utilizing innate syntactic representation in PLMs.

Our contributions are three-fold: (1) We propose a novel mechanism SSR, which learns to associate syntactic tree information with PLM representation. (2) We propose the Syntax-Reinforced Event (SRE) model, which employs SSR and features syntax-aware sentence representations for event extraction. (3) Experimental results on both sentence-level and document-level EE benchmark datasets show that our proposed SRE model achieves state-of-the-art performance in F1, with substantial improvements in recall for document-level EE tasks.

2 Related Work

Event Extraction is challenging as it concerns more than just binary relations. Among efforts to tackle the task on the sentence level, syntactic knowledge has always been relevant (Xiang and Wang, 2019). Early approaches to EE are feature-based methods (e.g., Li et al. (2013); McClosky et al. (2011)), in which syntactic features such as dependency tree parsing play a significant role. They are replaced by representation-based methods, which rely on powerful pre-trained language representations, from Word2vec to large language models (Xiang and Wang, 2019; Xu et al., 2024). Existing studies that explore syntactic enhancement can be largely divided into three categories: (1) using syntactic knowledge as auxiliary information for encoding, e.g., Chen et al. (2015); Nguyen et al. (2016), (2) proposing novel architectures using parsed dependency, e.g., Sha et al. (2018); Liu et al. (2018), (3) using syntactic trees to build graphs and performing graph neural network learning, e.g., Liu et al. (2018); Yan et al. (2019); Lai et al. (2020); Cui et al. (2020); Xie and Tu (2022). However, most of these studies rely on external parsers, which lead to error propagation[1]. Recently, probing studies on PLMs have found that syntactic structures can be emergent and intrinsic to language representation (Hewitt and Manning, 2019; Maudslay et al., 2020; Li et al., 2020; Newman et al., 2021). This indicates a potential alternative approach to the existing EE studies. In this paper, we propose a soft syntactic reinforcement mechanism to reinforce intrinsic syntactic knowledge learning. We use a human-annotated, task-independent parse tree bank for pre-training, instead of parsed dependency results of the task dataset from a trained parser.

Document-level EE, with argument entities scattering across sentences, has been formulated as converting the text containing event information into structured event description based on pre-defined templates, which was first defined as Template Filling in the MUC paradigm (Chinchor, 1992). State-of-the-art models include the following: MMR (Yang et al., 2022) is a multi-turn and multi-granularity, machine reading comprehension-based model. NST (Du and Cardie, 2020a) proposes a sequence tagging model that uses a gate mechanism to merge sentence and paragraph representations. TempGen (Huang et al., 2021) frames the task as a template generation task, incorporating a copy mechanism that takes the top-k important cross-attentions as copy distributions into BART. RICB (Zhou and Mao, 2022) leverages Redundant Information and Closed Boundary Loss.

Recent work in event extraction has also highlighted the potential of generative methods, particularly in few-shot and zero-shot learning scenarios (Xu et al., 2024; Ma et al., 2023; Huang et al., 2024). However, their performance in fully supervised event extraction, both at the sentence and document levels, often lags behind that of existing extractive methods. This is due to challenges including the complexity of accurately generating event structures and the tendency for generative models to produce hallucinations, raising concerns about their reliability in practical applications (Xu et al., 2024). In this work, we adopt the extractive approach and our model outperforms the state-of-the-art generative methods (Du and Cardie, 2020b; Liu et al., 2020; Hsu et al., 2022; Du et al., 2021; Huang et al., 2021).

[1]Additionally, some studies do not provide codes or have other reproducibility issues.

3 Proposed Approach

In this section, we first introduce the intuition of intrinsic syntactic encoding (Section 3.1). Then, we describe the proposed Soft Syntactic Reinforcement (SSR) mechanism (Section 3.2), and the Syntax-Reinforced Event (SRE) model that incorporates the SSR mechanism (Section 3.3).

3.1 Intrinsic Syntactic Encoding

Studies on factual knowledge in PLM probing suggested that syntax trees can be embedded implicitly in deep models' vector geometry. In other words, deep contextual models encode the parse tree information in their word representations. Hewitt and Manning (2019) demonstrated that a low-rank transformation can recover the parse trees from BERT representations, without using any syntactically annotated data as input or being supervised to reconstruct them. Inspired by studies along this line (Hewitt and Manning, 2019; Maudslay et al., 2020; Li et al., 2020; Newman et al., 2021), we aim to enhance the syntactic knowledge that is captured in PLM representation learning layers by aligning a linear transformation of the relevant weights to a human-annotated syntactic tree dataset. Subsequently, we use the consolidated weights to enrich the task-specific learning.

We construct the emergent syntactic structure bias in the form of dependency parsing trees which can be decomposed into syntactic *depth* and *distance*. Figure 2 illustrates an example of dependency parsing based on depth and distance (event triggers are shown in red and argument role labels are shown in blue or green).

Formally, syntactic depth refers to the syntactic tree edge path length from each word to the root. For example, in Figure 2, the word "*detonated*" is the root, which has a syntactic depth of 0. The first word "*Last*", is two edges away from the root, and thus its syntactic depth is 2. The syntactic depth vector d of a n-word sentence has the dimension of n.

Syntactic distance refers to the syntactic tree edge path length between each pair of words. For example, the word "*injuring*" has "*detonated*" as its parent, so the syntactic distance between "*injuring*" and "*detonated*" is 1. Similarly, the distance between "*injuring*" and its child "*soldiers*" is also 1. The syntactic distance matrix of a n-word sentence has the dimension of $n \times n$.

Given a syntactic tree T with the root node (word) r, the syntactic distance d_T measures the number of edges between each pair of nodes. The syntactic depth of node u is $d_T(u, r)$. For neighboring nodes u and v, the syntactic distance $d_T(u, v) = 1$.

3.2 Soft Syntactic Reinforcement

With the assumption of intrinsic syntactic encoding, we devise a soft syntactic reinforcement (SSR) mechanism to inject syntactic knowledge into the model. Unlike existing studies (e.g., Pouran Ben Veyseh et al. (2020)), we do not use any external parser to generate syntactic structures on task-specific data. Instead, we learn the transformation matrices that project PLM representations to new syntax-relevant spaces, and use them to obtain soft syntactic representation of input tokens to reinforce the overall representation learning for the EE task.

Following Hewitt and Manning (2019), we use linear transformation of a PLM-based word representation space to induce a syntax-salient representation space via dependency tree reconstruction. Let $\mathcal{M}$ be an encoder model that takes in a sentence of n words $S = \{w_i\}, i \in \{1, \ldots, n\}$, and produces a sequence of vector representations $\boldsymbol{h} = \{\boldsymbol{h}_i = \mathcal{M}(w_i)\}, i \in \{1, \ldots, n\}$. Let T be the annotated dependency parse tree of the sentence S. We construct a transformation matrix $\boldsymbol{B}_{dep}$ that maps $\boldsymbol{h}$ to the depth vector $\boldsymbol{d}_T = \{d_T(w_i, w_r) \mid w_i, w_r \in S, w_r \text{ is the root of } T\}$. Similarly, we construct $\boldsymbol{B}_{dist}$ to map to the distance matrix $\boldsymbol{D}_T = \{d_T(w_i, w_j) \mid w_i, w_j \in S\}$. We have $\boldsymbol{d}_T \in \mathbb{N}^n, \boldsymbol{D}_T \in \mathbb{N}^{n \times n}$. The transformation matrices $\boldsymbol{B}$ characterizes syntax-salient normalization of the representation space learned by the language model. To force the SSR module to select syntax-relevant dimensions, we first define the squared depth norm of the i-th word as:

$$\|\boldsymbol{h}_i\|_B^2 = (\boldsymbol{B}_{dep}\boldsymbol{h}_i)^\mathsf{T}(\boldsymbol{B}_{dep}\boldsymbol{h}_i) \tag{1}$$

and the squared distance between the i-th and j-th words in the sentence S as:

$$\mathrm{d}_B(\boldsymbol{h}_i, \boldsymbol{h}_j)^2 = [\boldsymbol{B}_{dist}(\boldsymbol{h}_i - \boldsymbol{h}_j)]^\mathsf{T}[\boldsymbol{B}_{dist}(\boldsymbol{h}_i - \boldsymbol{h}_j)] \tag{2}$$

Then, we pre-train the SSR module to minimize the following loss functions, Equation (3) for depth-based transformation and Equation (4) for distance-based transformation:

$$\min_{B_{dep}} \sum_S \frac{1}{n} \sum_i |\mathrm{d}_T(w_i, w_r)^2 - \|\boldsymbol{h}_i\|_B^2| \tag{3}$$

$$\min_{B_{dist}} \sum_S \frac{1}{n^2} \sum_{i,j} |\mathrm{d}_T(w_i, w_j) - d_B(\boldsymbol{h}_i, \boldsymbol{h}_j)^2| \tag{4}$$

We use the Penn Tree Bank dataset, a manually annotated syntactic dependency dataset (Marcus et al., 1993), for this stage.

3.3 Model Architecture

This subsection describes the architecture of our proposed soft Syntax-Reinforced Event (SRE) model, which integrates the SSR mechanism described in Section 3.2 into the baseline architecture.

There are two considerations in choosing the baseline architecture. First, based on the recent EE evaluation framework, TextEE (Huang et al., 2024), we find empirically that after carefully fine-tuning the PLMs, the baseline's performance on the task is comparable to that of the state-of-the-art models. Second, the selected baseline models should be relatively simple in terms of components, which makes analysis easier and benefits future studies. For Sentence-level EE (SEE), we use the architecture which consists of a pre-trained PLM encoder, such as RoBERTa (Liu et al., 2019b), and a CRF decoder (Figure 3), as our baseline. For document-level EE, we adopt the Multi-Granularity Reader (MGR) model (Du and Cardie, 2020a) as the baseline.

We now formally describe applying Soft Syntactic Reinforcement to the baseline models. Given an n-word input sentence $S = \{w_1, \ldots, w_n\}$, the model predicts $\{y_i\}, i \in \{1, \ldots, n\}$, as follows:

$$\boldsymbol{h}_{1:n} = \mathsf{PLM}(w_{1:n}) \tag{5}$$

$$y_{1:n} = \mathsf{CRF}(\boldsymbol{h}_{1:n}) \tag{6}$$

To incorporate the soft syntactic reinforcement, for example, using the depth-based matrix $\boldsymbol{B}_{dep}^k$ learned based on the k-th layer encoder representation, we get $\tilde{e}_i$, the transformed word representations with syntactic tree information[2] as follows:

$$\tilde{e}_i = \boldsymbol{B}_{dep}^k \boldsymbol{h}_i^k \tag{7}$$

[2]We call SSR "soft" because $\tilde{e}_i$ is obtained in a similar way as soft attention (Bahdanau et al., 2014), i.e., by gathering the reinforced syntactic information across multiple dimensions, rather than selecting the "most syntactically relevant" dimension.

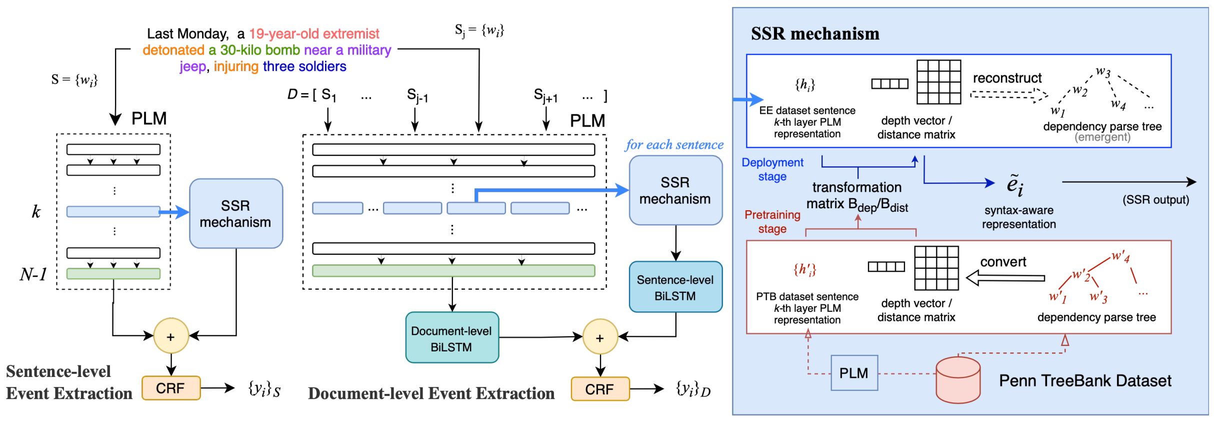
Last Monday, a 19-year-old extremist detonated a 30-kilo bomb near a military jeep, injuring three soldiers
S = {w_i}
S_j = {w_i}
PLM
k
N-1
SSR mechanism
+
CRF
{y_i}_S
Sentence-level Event Extraction
D = [S_1 ... S_j-1 S_j+1 ...]
PLM
for each sentence
SSR mechanism
Sentence-level BiLSTM
Document-level BiLSTM
+
CRF
{y_i}_D
Document-level Event Extraction
SSR mechanism
{h_i}
EE dataset sentence k-th layer PLM representation
depth vector / distance matrix
reconstruct
w_1 w_2 w_3 w_4 ...
dependency parse tree (emergent)
Deployment stage
transformation matrix B_dep/B_dist
Pretraining stage
ẽ_i
syntax-aware representation
(SSR output)
{h'_i}
PTB dataset sentence k-th layer PLM representation
depth vector / distance matrix
convert
w'_1 w'_2 w'_3 w'_4 ...
dependency parse tree
PLM
Penn TreeBank Dataset

Figure 3: Proposed SRE model architecture for Sentence-level and Document-level Event Extraction.

where $\boldsymbol{h}_i^k$ is the k-th layer PLM representation of the i-th word in a sentence[3]. For the distance-based SSR, $\boldsymbol{B}_{dep}^k$ is replaced by $\boldsymbol{B}_{dist}^k$ in this step. We then combine $\tilde{e}_i$ with the last-layer PLM representation $\boldsymbol{h}_i^{K-1}$ to get a fused representation $\boldsymbol{r}_i$:

$$\boldsymbol{g}_i = sigmoid(\mathsf{W}_1\boldsymbol{h}_i^{K-1} + \mathsf{W}_2\tilde{e}_i + b) \quad (8)$$

$$\boldsymbol{r}_i = \boldsymbol{g}_i \odot \boldsymbol{h}_i^{K-1} + (1 - \boldsymbol{g}_i) \odot \tilde{e}_i \quad (9)$$

where K is the total number of PLM layers, $\odot$ represents element-wise product. The fused representation $\boldsymbol{r}_{1:L}$ with $\boldsymbol{B}_{dep}$ reinforcement is the input of CRF, replacing $\boldsymbol{h}_{1:L}$ in Equation (6). For SRE with both distance and depth reinforcement, we fuse the distance-based $\tilde{e}_i$ and depth-based $\tilde{e}_i$ using the same gating function in Equations (8) and (9), and then fuse its outputs with $\boldsymbol{h}_i^{K-1}$.

For document-level EE, as shown in Figure 3, on top of the baseline architecture, we deploy a sentence-level Soft Syntactic Reinforcement before a sentence-level BiLSTM, and do not introduce SSRs beyond the sentence level.

Formally, given a d-sentence document $D = \{S_1, \ldots, S_d\} = \{w_1, \ldots, w_n\}, j \in \{1, \ldots, d\}$, where w_i refers to the i-th word token in D and n is the text length, the model first encodes each word token in the same way as Equation (5). Then, same as in Equation (7), the syntax-aware representation $\tilde{e}_i$ is generated via an SSR mechanism from $\boldsymbol{h}_i^k$, which is the k-th layer PLM representation of the i-th word in a sentence. While SSR outputs $\{\tilde{e}_i\}$, for all tokens in the sentence are passed through a sentence-level BiLSTM, the sentence's last layer PLM outputs $\boldsymbol{h}_i^{K-1}$, are passed through a document-level BiLSTM together with other sentences' PLM outputs, to infuse document context. Finally, similar to Equations (8) and (9), a gating function combines the BiLSTM outputs of each word with a learnable gating factor $\boldsymbol{g}_i$, followed by a CRF to predict $\{y_i\}$.

4 Sentence-level Event Extraction[4]

4.1 Experimental Setup

Datasets We conduct evaluations on three benchmark datasets from the TextEE framework (Huang et al., 2024): ACE05, CASIE, and PHEE. ACE05 is the most widely used benchmark dataset for SEE. The documents contain event annotations for 33 types of news events from 6 news sources. We evaluate on its English subset. Cyber Attack Sensing and Information Extraction (CASIE) is a cybersecurity domain event dataset that defines 5 event types: Databreach, Phishing, Ransom, Discover, and Patch. Pharmacovigilance Event Extraction (PHEE) dataset contains over 5000 annotated events in medical case reports and biomedical literature. For all three datasets, we follow the TextEE framework (Huang et al., 2024) for data preprocessing and splits. The detailed statistics of the datasets are summarized in Table 6 in Appendix A.

Evaluation Metrics We use micro-average F1 as the primary metric. Specifically, **Tri-C** scores a predicted trigger as correct if its offsets and type match a gold trigger. **Arg-C** scores a predicted argument as correct if its offsets, role and event type match a gold argument. **Arg-C+** scores a predicted argument as correct if its offsets and role match a gold

[3]The actual value of k is to be empirically determined based on the results from the development set.

[4]For details of Task Formulation and Implementation for both sentence-level and document-level EE task, please refer to Appendices B and C.

Model	ACE05			CASIE			PHEE		
	Tri-C	Arg-C	Arg-C+	Tri-C	Arg-C	Arg-C+	Tri-C	Arg-C	Arg-C+
DyGIE++ (2019)	<u>71.31</u>	56.01	51.81	44.72	36.39	29.53	70.42	**60.84**	**45.65**
OneIE (2020)	71.05	59.93	**54.70**	70.57	54.23	22.05	69.98	37.51	29.76
AMR-IE (2021)	71.09	<u>60.62</u>	54.62	<u>71.83</u>	10.19	2.79	68.93	43.04	32.44
EEQA (2020b)	70.04	55.28	50.36	42.79	35.14	26.23	70.29	40.40	32.02
RCEE (2020)	70.51	55.50	51.04	42.06	32.79	23.67	70.89	41.61	33.10
TagPrime (2023)	69.95	59.83	<u>54.64</u>	69.29	<u>61.03</u>	<u>49.07</u>	<u>71.14</u>	51.74	40.58
DEGREE-E2E (2022)	66.82	55.15	49.09	60.66	27.05	14.61	69.13	49.29	36.50
SRE (ours)	**72.18**	**60.98**	54.45	**72.13**	**63.70**	**51.36**	**72.54**	<u>54.98</u>	<u>43.49</u>

Table 1: Performance results for Sentence-level Event Extraction. The best performance for each column is highlighted in boldface and the second-best performance is underlined.

Model	**Tri-C**	**Arg-C**	**Arg-C+**
dbRNN-E2E†	69.6	57.2	50.1
RoBERTa baseline	70.10	58.98	52.55
RoBERTa hard-synt	69.93	58.71	52.96
Syntax-RoBERTa	68.85	58.38	52.75
SRE (ours)	**72.18**	**60.98**	**54.45**

Table 2: Performance on other syntax-based methods based on ACE05. † We use the reported results of the same setting from the original paper.

argument AND its corresponding trigger's offsets and type match the corresponding gold trigger.

Compared Models We compare our proposed SRE model with the following three kinds of state-of-the-art models: (1) pipeline models, which include **TagPrime** (Hsu et al., 2023); (2) joint learning models, which include **DyGIE++** (Wadden et al., 2019), **OneIE** (Lin et al., 2020) and **AMR-IE** (Zhang and Ji, 2021); and (3) generative models, which include **EEQA** (Du and Cardie, 2020b), **RCEE** (Liu et al., 2020), and **DEGREE-E2E** (Hsu et al., 2022).

To assess the performance impacts of our proposed mechanism as a syntax-based enhancement approach, we compare it with methods that utilize externally parsed syntactic information: (1) the state-of-the-art end-to-end EE model, **dbRNN-E2E** (Sha et al., 2018), which is based on a novel dependency bridge RNN structure, (2) the traditional hard syntactic approach (**RoBERTa hard-synt**), which uses dependency parsed features as learnable embeddings and (3) the state-of-the-art syntactic method to enhance Transformer-based models, **Syntax-RoBERTa** (Bai et al., 2021), which constructs syntactic masks based on parsed syntax trees. For fair comparison, RoBERTa hard-synt and Syntax-RoBERTa, and the PLM baseline are all based on RoBERTa-large, as that for our best SRE model. We implement BERT hard-synt using the state-of-the-art SpaCy (Honnibal and Montani, 2017) dependency parser. The experiments are conducted on ACE05, as it is the most commonly used SEE benchmark dataset.

4.2 Results

State-of-the-art Results As shown in Table 1, our proposed SRE model consistently outperforms other baselines on ACE05 with its Tri-C F1 of 72.18% and Arg-C F1 of 60.98%. Its performance on F1 for Arg-C+ is also comparable to the state-of-the-art models such as OneIE and TagPrime. On CASIE, we observe that our proposed SRE model performs better or is comparable to the current state-of-the-art models, with improvements of 0.3%, 2.67%, and 2.29% on Trig-C, Arg-C, and Arg-C+ F1, respectively. On PHEE, SRE achieves the best F1 score (72.54%) on Tri-C among all the models, and its argument classification outperforms all other models except DyGIE++. Overall, across all three datasets, we found that SRE consistently improves the baseline by large margins and achieves competitive results, especially on Tri-C.

Performance of Syntax-based Methods In Table 2, we compare our proposed SRE model against other syntax-based methods. The performance results show that our proposed SRE model based on RoBERTa outperforms the RoBERTa baseline with F1 improvement of 2.08%, 2.00%, and 1.90% on Tri-C, Arg-C and Arg-C+, respectively. SRE also performs better than dbRNN-E2E, RoBERTa

Model	Head Noun			Exact Match		
	P	R	F1	P	R	F1
Cohesion (Huang and Riloff, 2012)	57.80	59.40	58.59	-	-	-
NST (Du and Cardie, 2020a)	56.44	62.77	59.44	52.03	56.81	54.32
MMR (Yang et al., 2022)	**63.95**	58.71	61.19	**60.66**	55.34	57.87
SRE (ours)	61.93	**67.75**	**64.71**	56.84	**63.10**	**59.81**

Table 3: Performance comparison of our proposed SRE and the state-of-the-art models on Head Noun and Exact Match metrics.

Model	P	R	F1
Cohesion	58.38	39.53	47.14
NST	56.82	48.92	52.58
GRIT	64.19	47.36	54.50
TempGen	**68.55**	49.90	57.76
RICB	57.68	58.03	57.85
SRE (ours)	55.93	**61.84**	**58.95**

Table 4: Performance (%) comparison of our proposed SRE and the state-of-the-art models on CEAF-REE.

hard-synt, and Syntax-RoBERTa, across the three subtask evaluations. While we use our best SSR mechanisms for comparison, we also found that other SSR configurations can similarly improve on both BERT and RoBERTa (See Appendix D).

5 Document-Level Event Extraction

5.1 Experimental Setup

Dataset We conduct evaluations on the MUC-4 benchmark dataset (Sundheim, 1992). It contains 1,700 newswire texts with 5 types of event templates about terrorist events. Each template specifies multiple arguments of concern to a type of event. We follow prior work in data pre-processing and splits (Huang et al., 2021; Zhou and Mao, 2022). The detailed statistics of the datasets are summarized in Table 7 in Appendix A.

Evaluation Metrics We use micro-average Precision (P), Recall (R), and F1 based on Head Noun (**HN**) and Exact Match (**EM**) as the standard metrics. HN only evaluates predicted argument mentions based on their head noun, whereas EM requires predicted mentions to match the whole phrase of a gold one. To compare with more recent models, we also report results based on a new metric **CEAF-REE** (Du et al., 2021) F1, which encourages implicit coreference resolution.

Compared Models We compare our SRE proposed model with the following state-of-the-art models: Cohesion (Huang and Riloff, 2012), MMR (Yang et al., 2022) and NST (Du and Cardie, 2020a) on the old metrics, and Cohesion, NST, GRIT (Du et al., 2021), TempGen (Huang et al., 2021) and RICB (Zhou and Mao, 2022) on the new CEAF-REE metric.

5.2 Results

New State-of-the-art Performance Table 3 reports the performance comparison of our proposed framework and state-of-the-art methods on MUC-4 using the standard metrics. We observe that our models significantly improve both HN recall and EM recall compared to other methods. More specifically, our proposed SRE model achieves 67.75% in HN recall, which is 8.35% higher than the feature-based Cohesion method, and 4.98%-9.04% higher than the neural methods NST and MMR. It also achieves 63.10% in EM recall, which is 2.44%-6.29% higher than all the other methods.

On CEAF-REE, as shown in Table 4, our proposed SRE model achieves 58.95%, which is the highest F1 among all compared models. It is 1.10% higher than RICB, the second-best model. It also achieves exceptionally high recall, 61.84%, which is 3.79% higher than RICB.

Comparison with PLM baseline Our best document-level SRE model uses BERT-large as PLM. We conducted experiments to evaluate the improvement of SRE models over the baseline and found that depth+distance SSR achieves the highest F1 score of 58.95%, improving the PLM baseline for 1.61% on CEAF-REE F1. SREs with depth-based and distance-based SSR mechanisms have similar positive effects (See Appendix D).

Sentence-level EE	**Gold**	**OneIE**	**RoBERTa$_{\text{Large}}$**	**SRE**
Eight people ①, including *a pregnant woman* ② and *a 13-year-old child* ③ were **killed** ⓐ in Monday's Gaza raid.	ⓐTrg[ATTACK]	✓	✓	✓
	①Arg[TARGET]	✓	✓	✓
	②Arg[TARGET]	✓	✓	✓
	③Arg[TARGET]	NONE	NONE	✓

Document-level EE	**Gold**	**RICB**	**BERT$_{\text{Large}}$**	**SRE**
... ⟨2⟩ The massacre against the *Salvadoran Workers National Union Federation* ① (*FENASTRAS*) ② was planned in cold blood. ... ⟨4⟩ We have trustworthy information from our intelligence organs that this action was ordered by *Colonel Ponce* ③, that *Cristiani* ④ knew about it and approved it, and that it was carried out by *Colonel Elena Fuente* ⑤ as the head of the morbid death squad ... ⟨7⟩ Only a few days ago, *ARENA assassins* ⑥ tried to kill the *president of the Mortgage Bank*⑦, *Mr Mason* ⑧, for not following their orders. ...	ATTACK Event 1			
	①TARGET	✓	✓	✓
	②TARGET	✓	NONE	✓
	③PERPIND	✓	✓	✓
	④PERPIND	NONE	NONE	✓
	⑤PERPIND	NONE	NONE	✓
	ATTACK Event 2			
	⑥PERPIND	✓	✓	✓
	⑦NONE	✓	✓	TARGET
	⑧TARGET	NONE	NONE	✓

Table 5: Case studies for sentence-level and document-level EE between a SOTA model, the PLM baseline, and our proposed SRE model. For sentence-level EE, the triggers (Trg) are in bold with circled alphabetic labels and the arguments (Arg) are in italics with circled number labels. For document-level EE, sentence indices are in angle brackets. The argument candidates are in italics with circled number labels. '✓' indicates a correct prediction.

6 Case Studies

To better understand how our proposed SSR mechanism helps EE, we present two case studies based on the ACE05 and the MUC-4 development set, which are shown in Table 5[5]. For sentence-level EE, the sentence "*Eight people, including a pregnant woman and a 13-year-old child were killed in Monday's Gaza raid*" has the event trigger word "*killed*" (event type= CONFLICT:ATTACK) with the three corresponding arguments, which are in italics. While OneIE and RoBERTa$_{\text{Large}}$ fail to identify the third argument mention "*a 13-year-old child*" as a TARGET, our proposed SRE model can correctly identify it. Analysing the depth-based SRE model, we found that the predicted parse depths of the head words of the two argument mentions, "*woman*" and "*child*", are similar to one another, possibly providing a clue for the task. Our distance-based SRE model also correctly predicts the argument "*a 13-year-old child*". We observe that the distance-based SRE learns short distances between the indefinite articles "*a*", as well as the head words ("*woman*" and "*child*") of candidate arguments ① and ②. Moreover, the model learns short distances between the verb "*killed*" and each of the three arguments, i.e., "*people*", "*woman*" and "*child*".

[5]PERPIND and PERPORG are short forms of PERPETRATOR INDIVIDUAL and PERPETRATOR ORGANIZATION, respectively.

For document-level EE, we show the differences in the predictions of RICB, BERT$_{\text{Large}}$, and our proposed SRE model. Table 5 shows two events in this paragraph. For Event 1, RICB and BERT$_{\text{Large}}$ only detect "*Colonel Ponce*" as a PERPIND argument, whereas SRE successfully detects "*Cristiani*" and "*Colonel Elena Fuente*" as well. Besides, we can see that sentence ⟨4⟩ is very long. We observe that the SSR mechanism in SRE helps the model link "*this action*" with the following "*it*" pronouns, while action verbs "*ordered*", "*knew*", and "*carried out*" have similar estimated syntactic depths. This demonstrates the usefulness of the incorporation of syntactic knowledge. Similarly, for Event 2, SRE identifies "*Mason*" in sentence ⟨7⟩ as a TARGET argument, while RICB and BERT$_{\text{Large}}$ fail. Also, SRE predicts "*president of the Mortgage Bank*" as a TARGET argument. Although it is not a gold argument with the corpus, it is appositive to "*Mason*", and thus also a correct argument.

7 Conclusion

In this paper, we propose a Soft Syntactic Reinforcement mechanism for neural event extraction. To our best knowledge, this is the first work exploring mechanisms that enhance syntactic knowledge intrinsically captured by PLMs. Experiments on both sentence-level and document-level EE benchmark datasets show that our proposed method achieves state-of-the-art results and significantly improves recall for document-level EE.

8 Limitations

We did not investigate more complex SSR mechanisms. Probing studies have used more than linear transformation (White et al., 2021) and the emergent syntactic structures are also not limited to dependency parse trees. Constituency and other syntactic parsing structures are also valid choices for syntactic reinforcement (Arps et al., 2022). We chose to use dependency parse tree as it is most suitable for our research objective of syntactic reinforcement. It contains dependency relationships between words, which helps identification of mentions of key information for event extraction. Constituency trees, for example, focus on the hierarchical structure of phrases within a sentence and associate these phrases with certain parts of speech (e.g., noun phrase, verbal phrase), which is less directly beneficial for our purpose and may be more useful for tasks like machine translation. Future work may explore these alternative structures.

We also limited the syntactic pre-training data to Penn Tree Bank. The effects of pre-training on other corpora and other pre-training variations on EE performance remain to be investigated.

9 Acknowledgments

This research is supported by the Agency for Science, Technology and Research (A*STAR), Singapore. The authors thank Associate Professor Hui Siu Cheung for his guidance. The authors also thank the anonymous reviewers and the chairs of NAACL 2025 for their time and feedback.

References

David Arps, Younes Samih, Laura Kallmeyer, and Hassan Sajjad. 2022. Probing for constituency structure in neural language models. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 6738–6757, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Jiangang Bai, Yujing Wang, Yiren Chen, Yaming Yang, Jing Bai, Jing Yu, and Yunhai Tong. 2021. Syntax-BERT: Improving pre-trained transformers with syntax trees. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3011–3020, Online. Association for Computational Linguistics.

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.

Nancy Chinchor. 1992. MUC-4 evaluation metrics. In *Fourth Message Understanding Conference (MUC-4): Proceedings of a Conference Held in McLean, Virginia, June 16-18, 1992*.

Shiyao Cui, Bowen Yu, Tingwen Liu, Zhenyu Zhang, Xuebin Wang, and Jinqiao Shi. 2020. Edge-enhanced graph convolution networks for event detection with syntactic relation. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2329–2339, Online. Association for Computational Linguistics.

Xinya Du and Claire Cardie. 2020a. Document-level event role filler extraction using multi-granularity contextualized encoding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8010–8020, Online. Association for Computational Linguistics.

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

Xinya Du, Alexander Rush, and Claire Cardie. 2021. GRIT: Generative role-filler transformers for document-level event entity extraction. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 634–644, Online. Association for Computational Linguistics.

John Hewitt and Christopher D. Manning. 2019. A structural probe for finding syntax in word representations. 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 4129–4138, Minneapolis, Minnesota. Association for Computational Linguistics.

Matthew Honnibal and Ines Montani. 2017. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear.

I-Hung Hsu, Kuan-Hao Huang, Elizabeth Boschee, Scott Miller, Prem Natarajan, Kai-Wei Chang, and Nanyun Peng. 2022. DEGREE: A data-efficient generation-based event extraction model. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1890–1908, Seattle, United States. Association for Computational Linguistics.

I-Hung Hsu, Kuan-Hao Huang, Shuning Zhang, Wenxin Cheng, Prem Natarajan, Kai-Wei Chang, and Nanyun Peng. 2023. TAGPRIME: A unified framework for relational structure extraction. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12917–12932, Toronto, Canada. Association for Computational Linguistics.

Kuan-Hao Huang, I-Hung Hsu, Tanmay Parekh, Zhiyu Xie, Zixuan Zhang, Premkumar Natarajan, Kai-Wei Chang, Nanyun Peng, and Heng Ji. 2024. Textee: Benchmark, reevaluation, reflections, and future challenges in event extraction.

Kung-Hsiang Huang, Sam Tang, and Nanyun Peng. 2021. Document-level entity-based extraction as template generation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5257–5269, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Ruihong Huang and Ellen Riloff. 2012. Modeling textual cohesion for event extraction. In *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence*, AAAI'12, page 1664–1670. AAAI Press.

Diederik P. Kingma and Jimmy Ba. 2017. Adam: A method for stochastic optimization.

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 the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5405–5411, Online. Association for Computational Linguistics.

Huayang Li, Lemao Liu, Guoping Huang, and Shuming Shi. 2020. On the branching bias of syntax extracted from pre-trained language models. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4473–4478, Online. 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.

Ying Lin, Heng Ji, Fei Huang, and Lingfei Wu. 2020. A joint neural model for information extraction with global features. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7999–8009, Online. Association for Computational Linguistics.

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

Jian Liu, Yubo Chen, Kang Liu, and Jun Zhao. 2019a. Neural cross-lingual event detection with minimal parallel resources. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 738–748, Hong Kong, China. Association for Computational Linguistics.

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.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. Roberta: A robustly optimized bert pretraining approach.

Yubo Ma, Zehao Wang, Yixin Cao, and Aixin Sun. 2023. Few-shot event detection: An empirical study and a unified view. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11211–11236, Toronto, Canada. Association for Computational Linguistics.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Rowan Hall Maudslay, Josef Valvoda, Tiago Pimentel, Adina Williams, and Ryan Cotterell. 2020. A tale of a probe and a parser. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7389–7395, Online. Association for Computational Linguistics.

David McClosky, Mihai Surdeanu, and Christopher Manning. 2011. Event extraction as dependency parsing. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1626–1635, Portland, Oregon, USA. Association for Computational Linguistics.

Benjamin Newman, Kai-Siang Ang, Julia Gong, and John Hewitt. 2021. Refining targeted syntactic evaluation of language models. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3710–3723, Online. Association for Computational Linguistics.

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.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

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, Online. Association for Computational Linguistics.

Sebastian Ruder, Matthew E. Peters, Swabha Swayamdipta, and Thomas Wolf. 2019. Transfer learning in natural language processing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Tutorials*, pages 15–18, Minneapolis, Minnesota. 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. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1).

Beth M. Sundheim. 1992. Overview of the fourth Message Understanding Evaluation and Conference. In *Fourth Message Understanding Conference (MUC-4): Proceedings of a Conference Held in McLean, Virginia, June 16-18, 1992*.

David Wadden, Ulme Wennberg, Yi Luan, and Hannaneh Hajishirzi. 2019. Entity, relation, and event extraction with contextualized span representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5784–5789, Hong Kong, China. Association for Computational Linguistics.

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 the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1652–1671, Online. Association for Computational Linguistics.

Xiaozhi Wang, Ziqi Wang, Xu Han, Zhiyuan Liu, Juanzi Li, Peng Li, Maosong Sun, Jie Zhou, and Xiang Ren. 2019. HMEAE: Hierarchical modular event argument extraction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5777–5783, Hong Kong, China. Association for Computational Linguistics.

Jennifer C. White, Tiago Pimentel, Naomi Saphra, and Ryan Cotterell. 2021. A non-linear structural probe. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 132–138, Online. Association for Computational Linguistics.

Wei Xiang and Bang Wang. 2019. A survey of event extraction from text. *IEEE Access*, 7:173111–173137.

Zhipeng Xie and Yumin Tu. 2022. A graph convolutional network with adaptive graph generation and channel selection for event detection. In *AAAI Conference on Artificial Intelligence*.

Derong Xu, Wei Chen, Wenjun Peng, Chao Zhang, Tong Xu, Xiangyu Zhao, Xian Wu, Yefeng Zheng, Yang Wang, and Enhong Chen. 2024. Large language models for generative information extraction: A survey.

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 the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5766–5770, Hong Kong, China. Association for Computational Linguistics.

Hang Yang, Yubo Chen, Kang Liu, Jun Zhao, Zuyu Zhao, and Weijian Sun. 2022. Multi-turn and multi-granularity reader for document-level event extraction. *ACM Trans. Asian Low-Resour. Lang. Inf. Process.*, 22(2).

Zixuan Zhang and Heng Ji. 2021. Abstract Meaning Representation guided graph encoding and decoding for joint information extraction. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 39–49, Online. Association for Computational Linguistics.

Hanzhang Zhou and Kezhi Mao. 2022. Document-level event argument extraction by leveraging redundant information and closed boundary loss. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3041–3052, Seattle, United States. Association for Computational Linguistics.

A Datasets

Figure 6 and 7 summarize the detailed statistics for the sentence-level and the document-level datasets, respectively.

Dataset	Split	#Doc	#Inst	#Event	#Arg
ACE05	Train	481	16,887	4,325	6,527
	Dev	59	1,957	503	792
	Test	59	2,076	520	778
	Total	599	20,920	5,348	8,097
CASIE	Train	701	1,044	5,973	15,890
	Dev	149	220	1,252	3,318
	Test	149	219	1,244	3,367
	Total	999	1,483	8,469	22,575
PHEE	Train	2,897	2,897	3,011	15,456
	Dev	965	965	1,002	5,117
	Test	965	965	1,006	5,186
	Total	4,827	4,827	5,019	25,760

Table 6: Detailed statistics for the sentence-level datasets, ACE05, CASIE and PHEE, including the number of documents, instances, events, and arguments, with average counts across 5 data splits.

B Task Formulation

We formulate the sentence-level EE as a sequence tagging task, formally defined as follows: given a sentence $S = \{w_1, \ldots, w_L\}$, where w_i refers to the i-th word token and L is the sentence length, the model outputs a sequence of labels $Y = \{y_1, \ldots, y_L\}$, where each y_i indicates all the event *triggers* (text spans that depict events) and the corresponding event *arguments* (text spans that denote entities taking certain roles in an event).

We formulate the document-level EE task as follows: given an input document comprised of N sentences $D = \{S\}$, the DEE task aims to extract one or more structured events $Y = y_i$, where each event y_i with event type t contains a series of roles $(r_1^t, r_2^t, \ldots, r_n^t)$ filled by argument spans, n is the number of pre-defined roles for the event type t, $t \in T$ and $r \in R$, T represents the set of pre-defined event types, and R is the set of role categories.

C Implementation Details

For sentence-level EE, we implement the proposed model in Pytorch (Paszke et al., 2019). To be consistent with previous studies (Wang et al., 2020, 2019), we use the Stanford CoreNLP toolkit for sentence segmentation. The non-content part of

Split	#Doc	#Sent	#Arg
Train	1,300	18,967	2,551
Dev	200	3,112	483
Test	200	2,786	533
Total	1,700	24,865	3,567

Table 7: Detailed statistics for the document-level dataset, MUC-4.

the raw text (e.g., XML tags) is excluded from the input. We use the large, cased version of BERT and the base uncased version of RoBERTa. We find k =8 for base-size models and 16 for large-size models tend to give best results in most cases. We use Adam (Kingma and Ba, 2017) optimizer with the learning rate tuned around 3e-5 with a linear warm-up. The batch size is set to 24. We implement early stopping (patience = 5) and limit the training to 50 epochs. We apply a dropout of 0.9 for both dense and attention layers. The values of λ and α are manually tuned between [1, 200] and [0, 1], respectively. The best results are selected from 5 runs based on development set evaluation results.

For document-level EE, we implement our proposed model based on PyTorch and Hugging Face's Transformer library. For each experiment we finetune BERT-large-cased and $k = 16$ tends to give best results in most cases. The Bi-LSTM modules for both sentence-level and document-level contextualized learning have 3 layers each. We set the maximum sequence length as 200 for sentences and 512 for paragraphs. The average training runtime for each model is 3.5 hours. The models were trained for 15 epochs with a batch size of 5. We use SGD as the optimizer and pose learning rate decay. We implement early stopping with the patience of 3. The learning rates betweeen range {1.5e-4, 1.5e-2} were evaluated, of which the starting learning rate of 1.5e-3 performed best on the validation set.

All the experiments were run on single NVIDIA RTX A6000 and Quadro GV100 GPUs.

D Performance comparison of different SSR mechanisms

For Sentence-level EE, we conduct experiments to compare the performance of various SSR mechanisms on BERT and RoBERTa. As shown in Table 8, the performance results show that the depth-based SSR mechanism with the RoBERTa

PLM	dep.	dist.	Tri-C	Arg-C	Arg-C+
BERT			66.43	53.14	48.61
BERT	✓		69.13	55.53	49.96
BERT		✓	68.23	55.05	49.74
BERT	✓	✓	67.46	53.69	49.08
$\text{RoBERTa}_{\text{Large}}$			70.10	58.98	52.55
$\text{RoBERTa}_{\text{Large}}$	✓		72.18	60.98	54.45
$\text{RoBERTa}_{\text{Large}}$		✓	71.76	60.89	54.38
$\text{RoBERTa}_{\text{Large}}$	✓	✓	70.98	60.31	53.83

Table 8: Performance on different PLMs for SSR based on ACE05.

Model	F1	Δ F1
BERT-large baseline	57.34	-
SRE depth+distance SSR	58.95	+1.61%
SRE depth SSR	58.74	+1.40%
SRE distance SSR	58.55	+1.21%

Table 9: Performance comparison of different SSR mechanisms on CEAF-REE.

encoder achieves a state-of-the-art performance of 72.18% on F1, which is 2.08% higher than RoBERTa PLM. The distance-based SSR mechanism and the depth+distance SSR mechanism also improve performance on RoBERTa, with F1 on Tri-C of 71.76% and 70.98%, respectively. With the BERT encoder, SRE_{BERT} also achieve improved results over the BERT baseline, with 67.46%-69.13%, 53.69%-55.53%, and 49.08%-49.96% F1 on Tri-C, Arg-C and Arg-C+, demonstrating the effectiveness of our proposed mechanism.

As shown in Table 9, we also compare different SSR mechanisms for our SRE model for document-level EE. We found that depth+distance SSR achieves a high F1 score of 58.95%, which is 1.61% higher than the PLM baseline. The depth-based SSR mechanism obtains an F1 score of 58.74%, whereas the distance-based SSR mechanism obtains an F1 score of 58.55%. The performance improvements over the baseline are 1.40% and 1.21%, respectively. We observe that all three variations of SSR mechanisms improve the performance similarly for DEE.

E Empirical Computational Costs

To show that our proposed method is computationally efficient, we present the estimated computational costs based on our empirical experiments of pre-training on the PennTree Bank corpus and training and inference on the sentence-level ACE05 dataset, in Table 10. We will also release the trained model checkpoints with our code to facilitate further experimentation.

	Time Cost
Depth-based SSR Pre-training	3-4 hours
Distance-based SSR Pre-training	4-5 hours
SRE Training	4-6 hours
SRE Inference	2-3 minutes

Table 10: Estimated times to complete different stages of our experiments.