Neural Processing Letters (2022) 54:227–249
https://doi.org/10.1007/s11063-021-10627-2

SynSeq4ED: A Novel Event-Aware Text Representation Learning for Event Detection

Tham Vo[1]

Accepted: 16 August 2021 / Published online: 6 September 2021

Abstract
Event detection (ED) is considered as an important task in natural language processing (NLP) which effectively supports to specify instances of multi event types which are mentioned in text. Recent models adopt advanced neural network architectures, such as long short-term memory (LSTM), graph convolutional network (GCN), etc. to capture the sequential and syntactical representations of texts for leveraging the performance of ED. However, recent neural network-based models neglect to sufficiently perverse both sequential comprehensive meanings as well as syntactical co-referencing relationships between words in the sentences. In this paper, we proposed a novel integration of GCN-based textual syntactical encoder and pre-trained BERT sequential embedding with event-aware masked language mechanism, called SynSeq4ED. In our SynSeq4ED model, we formally present a joint text embedding framework which enable to effectively learn the deep semantic representations of event triggers and arguments by introducing a combination of integrated pre-trained BERT with event-aware masked language strategy and GCN-based syntactical co-referencing text encoding mechanism. The achieved text representations by SynSeq4ED model are then used to improve the performance of multiple tasks in ED, including multiple event detection (MED), few-shot learning event detection (FSLED). Extensive experiments in benchmark datasets demonstrate the effectiveness of our proposed SynSeq4ED model in comparing with recent state-of-the-art baselines.

Keywords Event detection · GCN · BERT · Attention · Masked language model

1 Introduction

In general, event detection is an important downstream task of NLP [1, 2] domain which belongs the to the textual information extraction area. The proposed ED models support to detect event triggers and related arguments from a given text, then categorize them into different types. From the past, extracting events from texts is considered as a challenging task due to the need of thorough natural language analysis and understanding. Traditionally, most

✉ Tham Vo
thamvth@tdmu.edu.vn

[1] Thu Dau Mot University, Binh Duong, Vietnam

of designed ED techniques is considered as a supervised-learning approach which usually depend on existing labelled/annotated datasets which are manually constructed. Constructing these annotated datasets are considered as the laborious as well as inflexible for multiple disciplines. Hence, these traditional models for ED task encounter several challenges related to the difficulties of insufficient training data burdensome and the effectiveness of supervised-learning process. In realistic implementation, ED task task has common challenges which can be categorized into two aspects, which are: event role overlapping and unsufficient training data hinders. For the event role overlapping aspect in ED task, an event detection model should be able to precisely recognize mutiple event triggers with their specifice event types and the associated arguments of these identified events corresponding with the roles. This aspect seems challenging due to a fact is that multiple events might exist in a same textual content (e.g., sentence). Moreover, in this aspect, the ED task also faces a difficulty of many same event might occur in the forms of multiple trigger descriptive information/expressions and these event descriptive information might also be used to demonstrate the other events in different contexts. Thus, it leads to hinders for accurately analyzing, detecting and classifying event triggers and arguments into their types. For the unsufficient training data aspect, most of recent machine learning based techniques for event trigger/argument extraction are designed under the supervised/semi-supervised learning paradigms which majorly rely on lots of annotated data for producing efficient event trigger/argument classifiers. There is no doubt that constructing a large accurate amounts of training data is considered as the laborous and time-consuming task. The major dependence on training set also leads to hinders related to the capabilities of model expansion and self-learning/producing additional event for extra training process. Previous works in ED are typically designed to employ the supervised mechanism learning (e.g. SVM [3], neural networks [4]) on hand-crafted feature engineering process. However, the main problem of this approach is that they can't perform well on the circumstance of insufficient annotated training set and complex event trigger/argument overlapping. Moreover, the hand-crafted feature engineering for ED task is considered as a manual process which require lots of linguistic intuition as well as NLP-specific domain expertise. In other words, the hinders on feature engineering and textual representations are the main challenging issues of traditional ED-based techniques. Thus, researchers have extensively sought for alternative text representation learning methods for sufficiently preserving the relationships of event triggers/arguments and their associated textual contexts.

Recently, with the raise of deep learning [5], serveral studies have proposed to take advantages of deep learning-based architectures, such as: LSTM [6, 7, 8], convolutional neural network (CNN) [9, 10, 11], etc. to capture the sequential latent representations of words which directly reflect the relationships between texts and their associated informative contexts for effectively dealing with ED task. In CNN-based models for event extraction, the input word embeddings of a given sentence are fed into a multi-layered convolution network with different pool strategies (max pooling, dynamic multi-pooling [10], etc.) and then classified at the output layer with softmax function. However, the CNN-based models for event extraction task also encounter several limitations related to the efficiency in the neural network propagation processes for the event detection and the argument identification procedures. Furthermore, the CNN-based approach for ED task normally takes the concatenations of input word embedding vectors for constructing a completed sentence-level embedding matrix which is considered as unable to capture the sequential relationship structure between words in a given sentence. To overcome this limitation, multiple recurrent neural network (RNN)-based techniques have been proposed to effectively capture the consecutive dependent representations of words by taking the input words embeddings according to their sequential orders in a given sentence. Such as successes of Chen et al. [8] in proposing the use of

bidirectional LSTM (Bi-LSTM) architecture for jointly preserving the sequential latent representations of texts in both forward and backward directions. However, RNN-based baselines for ED task also has a limitation in preserving the syntactical long-range word dependency at the context of sentence/document-level. Recent well-known works of Nguyen et al. [12] which proposed the use of multi-layered GCN-based textual encoder to learn the long-range syntactical relationship representations of words for effectively identifying the event triggers and types in a sentence. From remarkable achievements of Nguyen et al., several graph-based neural network based techniques have been proposed to deal with different downstream subtasks (multiple event detection [13], few-shot event detection [14], etc.) in the event analysis and extraction domain.

Existing challenges and our motivations. Unfortunately, recent graph neural network based models for ED also have a limitation in considering all potential consecutive n-grams of given word embeddings in a sentence/document which might insufficient for alleviating unnecessary and noisy informative representations of words for ED task. Moreover, the relationships between events which are expressed in natural language texts are not just dependent on how implicit or explicit the events are mentioned in sequential representations of words but also mostly depend on the associated arguments and the syntactical representations as well. It requires a thorough linguistic comprehensive understanding between the particular linguistic expression of expressed events as well as the semantic contextual arguments which are established in the texts. In addition, in previous graph-based textual embedding techniques [12, 13, 14], there is no obvious solution for the ED task-specific joint representation learning between the sequential and structural syntactical embeddings of words in a given sentence/document. To deal with these limitations, in this paper we proposed a novel pre-trained BERT-based [15] textual representation learning model for ED, called SynSeq4ED. Our proposed SynSeq4ED model take the advantages of pre-trained BERT architecture with the defined entity-aware masked language mechanism for the fine-tuning process which enables to learn the additional information from the whole sentence/document structure, along with its semantic n-grams consecutive latent features. Figure 1 illustrates the overall architecture of our proposed SynSeq4ED model.

In the first step, to capture the syntactical and co-referencing relationship between words, we apply the multi-layered graph-based convolution archiecture over the parsed syntactic and co-referencing text graphs to effectively capture the word representations in different immediate neighborhood contexts. To jointly learn the word representations of multi-typed text graphs of a given sentence/document as the heterogeneous networks, we applied the graph tensor [16] propagation learning strategy. Then, the achieved word embedding vectors in previous step are used as the inputs for the pre-trained BERT model with the event-aware masked language mechanism to effectively deal with the event extraction downstream subtasks. By using the pre-trained BERT model in the process of text sequential representation learning, the learnt word embeddings can be enriched with the external contexts of words which enables to facilitate the deep understanding of both syntactic and consecutive structures of the whole setence/document. Thereby, it is promising to produce better result on the event detection task. To sum up, our contributions in this paper can be summarized as three folds, which are:

- First of all, we proposed a novel tensor graph-based textual embedding technique which enables to joinly capture the long-range dependent syntactic and co-referencing structures of words in a given sentence/document. To do this, we apply different GCN-based architectures to simultaneously learn the neighborhood aggregation of immediate word nodes in different constructed text graphs. To extract the syntactic and co-referencing relationships

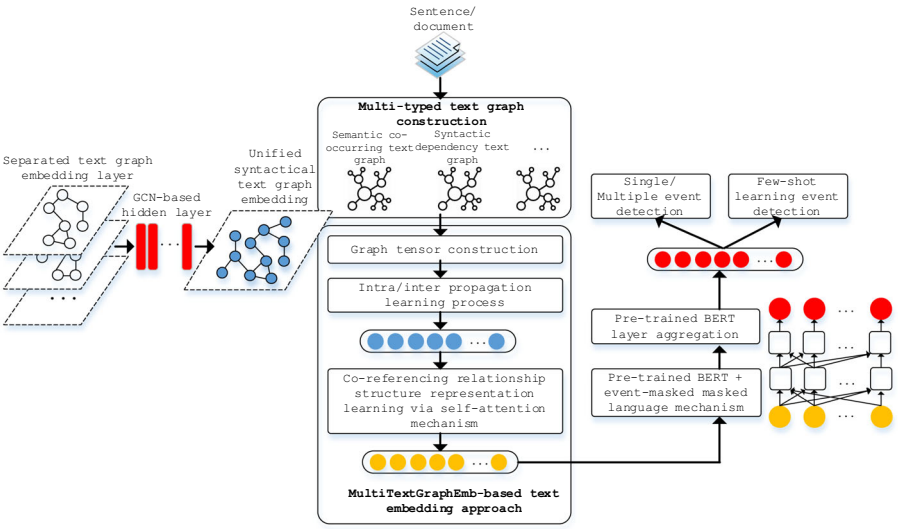


Fig. 1 The overall architecture of our proposed SynSeq4ED model

between words in each sentence/document, we mainly apply the Stanford CoreNLP library [17]. This proposed graph tensor-based joint dependent syntactic and co-referencing relationship textual embedding strategy which is considered as our main contribution in this paper is called as: MultiTextGraphEmb.

- Next, the word embeddings which are learnt from the MultiTextGraphEmb-based textual embedding technique is then fed into a pre-trained BERT model with a custom event-aware masked language machanism to learn the sequential relationships between words at the sentence-level context. During the training process, the pre-trained BERT will be forced to predict hidden event words at random masked position in a given sentence/document. The final achieved word embedding vectors are finally used to facilitate multiple primitive subtasks of event dection domain, including multiple event/argument detection and few-shot learning event detection.
- To demonstrate the effectiveness of our proposed SynSeq4ED model, we conducted extensive experiments in several benchmark datasets, including: ACE-2005 and TAC-KBP-2015. Comprehensive experimental outputs in standardized datasets present the outperformance of our proposed model in comparing with recent state-of-the-art baselines for ED task.

The left parts of our paper are organized into 5 sections. In the next section, we briefly present about general approaches and recent well-known studies which are related to event extraction area. In the third section, we formally provide ED related background concepts and common notations which are used in our paper. In the fourth section, we demonstrate the main ideas of our proposed SynSeq4ED model, detailed descriptions of our methodology and implementation. In the fifth section, we provide extensive experiments of multiple event extraction subtasks by different models in benchmark datasets. In this section, we also give careful discussions about the experimental results as well as further empirical studies on our model's configuration parameters. Finally, we conclude our achievements in the proposed SynSeq4ED model as well as provide some possible improvements for future works.

2 Related Works

In general, the event extraction system is designed to detect the existence of an event reported in text as well as event-related information (e.g., trigger, argument, etc.). Strictly following the guidelines of ACE [18], there are important terminologies of ED task, which are: event mention, trigger, argument and argument role. For the event mention, it might be a phase/sentence which describes specific related events, triggers and associated arguments. The event trigger is a word or compound word which indicates/expresses the occurrence of an event (typically a verb or noun). The event argument is a mentioned entity or the temporal event expression which is considered as the participant/attribute of an event with a specific role. The argument role is used to specify the relationship between an event and it associated arguments. Event extraction from texts has attracted researchers in the last decades due to its potential applications in multiple NLP's domains. There are traditional and successful proposed models for ED task have deeply adopted the combination of manual annotated data construction and hand-crafted feature engineering methods [18, 19, 20]. Recent work of Liu et al. [20] which applies the probabilistic reasoning model to classify the event using latent features from text. However, hand-crafted feature engineering-based approach encounters a major limitation of high-effort for the training data annotation process as well as sparsity in the textual representation. In the last couple of years, we witnessed the raise of advanced deep neural network architectures which demonstrates effectiveness in multiple downstream tasks of NLP domain, including the ED task. many deep neural architectures have been utilized to improve the performance of ED task such as GRU, LSTM and CNN. Such as well-known works of Nguyen et al. [9] which presented the use of CNN for effectively learning the textual latent features based on the continuous word embedding matrices which has been proven to be efficient for ED. Inherited from the success of [9] to that, Chen et al. [10] proposed a novel integrated CNN-based dynamic max-pooling strategy for separately evaluating different components of a given sentence via the designed dynamic multi-pooling extraction layer which enables to capture both lexical-level and sentence-level latent feature representation. To effectively preserve the n-gram sequential dependency features of words in a given sentence/document, Nguyen et al. [12] proposed the use of GCN-based textual embedding mechanism over the syntactical text graph of a given sentence/document. The proposed model of Nguyen et al. [12] demonstrates the capability of efficiently capturing the long-range dependent latent features between words for improving the performance of ED task. Considering GCN as the key factor for greate achivements in ED task, recent works [13, 14] have utilized the GCN-based syntactic text embedding technique for hanlding multiple event detection and few-shot learning event detection subtasks. On the other site, to effectively preserve the consecutive relationship features between words in a given sentence/document, several RNN-based architectures, such as GRU, LSTM, Bi-LSTM, etc. have been utilized in ED-based models. Recent well-known studies of Chen [8] in using Bi-LSTM-based architecture for exploiting the sequential dependency of words following their continuous forward and backward orders in a given sentence/document. Similar to that, Sha et al. [21] proposed a novel dependency bridge RNN-based approach, called as (dbRNN) which jointly learns the representations of syntactic dependency and sequential orders between words for ED task. Also taking advantage of dependent tree analysis in text, Zhang W. et al. proposed an approach [22] of using Tree-LSTM [23] based mechanism to effectively learn the word embeddings from the original parsed dependency tree via syntactic dependency analysis strategy for event detection in Chinese language.

However, in practical implementation, the event detection task still remained existing challenges due to the complexity of multiple semantic word embedding requirement for various contexts. Despite many recent successes in event extraction area, there is no work has studied the aspects of rich external contextal representation of texts for ED task. Moreover, recent GCN-based works [12, 14] for ED also mainly focus on preserving syntactic structure of texts for leveraging the performance ED and there is no obvious combination between the syntactical structural features with sequential relationships of words during the representation learning process.

3 Preliminaries and Problem Formulation

In this section, we formally present problem formulation of event detection task and background concepts related to textual embedding/representation learning via GCN-based and BERT-based architectures. Normally, the event detection task is defined as a multi-class classification problem where an ED-based trained classifier is used to predict whether each word/compound word in a given sentence/document should be the event triggers, metioned event and the associated arguments.

Definition 1 Event Detection (ED) is traditionally defined as a multi-class classification task with a ED-based trained classifier, denoted as a mapping function: $f_{ED}(.)$ which supports to assign proper classes as a set of (m) pre-defined event types, as: $\mathscr{C} = \{1, 2, \ldots, m\}$ (ACE-2005 dataset has total 33 event types), for a set of words/compound words in a given sentence with (n) length, denoted as: $\mathfrak{z} = \{w_1, w_1, \ldots, w_n\}$. Thus, the overall event extraction process can be simply defined as a mapping process, denote as: $f_{ED}(\mathfrak{z}) \rightarrow \{1, 2, \ldots, n\}$, where: $_i, i \in \mathscr{C}$ is the label of a specific i^{th} word in a given sentence $(\mathfrak{z})$. To learn parameters of $f_{ED}(.)$ function, multiple supervised learning methods (e.g. out-of-the-shelf classification algorithms, neural network, etc.) are applied on the annotated/training dataset, denoted as: $\mathscr{T} = \{\langle\{1, 2, \ldots, n\}|\{e_{w_1}, e_{w_2}, \ldots, e_{w_n}\}\rangle_t\}_{t=1}^{|\mathscr{T}|}$, where: e_{w_i} presents for the latent representation vector of a specific i^{th} word via different textual embedding technique (see definition 2).

Definition 2 TextualTextual. Embedding/Representation Learning: is considered as the main trend for most of advanced proposals in NLP domain, recently. The general textual or word representation learning is a technique which support to transform the word(w)/sentence$(\mathfrak{z})$/document() from the complex high-dimensional space into the fixed d-dimensional representation vector space. Normally, textual embedding technique is defined as a mapping function, denoted as: $f_{\text{text\_emb}}(w|\mathfrak{z}|) \rightarrow (e_w|e_\mathfrak{z}|e)$, where: $e \in \mathbb{R}^{1\times d}$ is the embedding vector of given word/sentence/document. There are primitive well-known word embedding models, such as: Word2Vec [24], GloVe [25], etc.

Definition 3 Graph Convolutional Network (GCN) [26]: is one of the most well-known network/graph-based structure representation learning techniques which is introduced by Kipf, T. et al. recently. The GCN is developed upon the idea of deep convolutional neural network (CNN) architecture which learn the node representations by the contextual neighborhood aggregation and multi-layered graph-based spectral propagation mechanisms. Normally, giving a graph-based structure, as: $G = (V, E)$, the GCN-based network representation learning process of G is defined as a deep multi-layered neural network architecture, where each specific k^{th} GCN-based layer is generally defined as: $H^{[k+1]} = f_{act}(W^{[k]}H^{[k]}\widehat{A})$, with: $f_{act}(.)$,W, H and $\widehat{A}$ are the activation function, weighting parameter matrix, current

hidden state matrix and normalized adjacency matrix version of a given graph-based structure (G), respectively.

Definition 4 Pre-trained BERT model [15] is the most well-known pre-trained language model which is considered as the sentence-level textual representation learning technique. BERT support to effectively capture the deep semantic meanings of words dynamically in consideration of their surrounding contexts. The pre-trained BERT model is a trained version with large-scale text corpora for general purposes which is considered as capable for carrying external rich-semantic contexts of words in different NLP's tasks for specific language. In general, the pre-trained BERT model can support to learn the whole sequence of sentences, as: $\mathfrak{z} = \{\mathfrak{z}_i\}_{i=1}^{|\mathfrak{z}|}$ and produce a set of hidden state vectors, denoted as: $f_{\text{BERT}}(\mathfrak{z}) \rightarrow \{\overrightarrow{h_1}, \overrightarrow{h_2}, \ldots, \overrightarrow{h_{|\mathfrak{z}|}}\}$, where: $\overrightarrow{h_i}$ is the hidden state vector of a specific i^{th} sentence which carries out the rich-semantic contextual relationship representation of a given sentence with the others.

Recent deep learning-based ED studies have mainly concentrated on how to find effective methods for transforming a set of given words in a sentence into embedding vectors and still keep latent contextual relationship information between words. Multiple advanced deep neural network-based architectures as well as auto-encoding mechanisms, such as: LSTM, CNN, GCN (Definition 3), pre-trained language models (e.g., ELMo, BERT—Definition 4, etc.) have been applied to achieve significant performances in ED task. Among deep neural network mechanism, GCN is mostly used due to its simplicity and effectiveness in capturing the long-range syntactical relationship between words for leverage the semantic contextual representation of text in multiple downstream subtasks of event extraction domain. However, recent GCN-based syntactical embedding models still have been considered as unable to jointly learn the syntactical and consecutive relationships between words in different contexts. Thus, to overcome this limitation, in this paper, we proposed a novel approach of textual embedding strategy which is aimed to leverage the performance of ED task by taking the advantages of pre-trained BERT with event-aware masked language mechanism which is fully integrated with the GCN-based syntactical and co-referencing relationship embedding approach. For further reference purpose, Table 1 provides all notations and math symbols which are commonly used in our paper.

4 Methodology and Implementations

In this section, we demonstrate detailed descriptions about the ideas, methodology and implementation of our proposed SynSeq4ED model. First of all, we present a novel integrated syntactical and co-referencing relationship textual embedding, called as MultiTextGraphEmb. Our proposed MultiTextGraphEmb texual embedding method supports to transform the discrete word distributions in each sentence into the contineous fixed d-dimensional vector space. Then, the achieved rich-semantic word representations are then used to feed into a pre-trained BERT model with the event-aware masked language mechanism to faciliate the ED task.

Table 1 List of used notations & descriptions

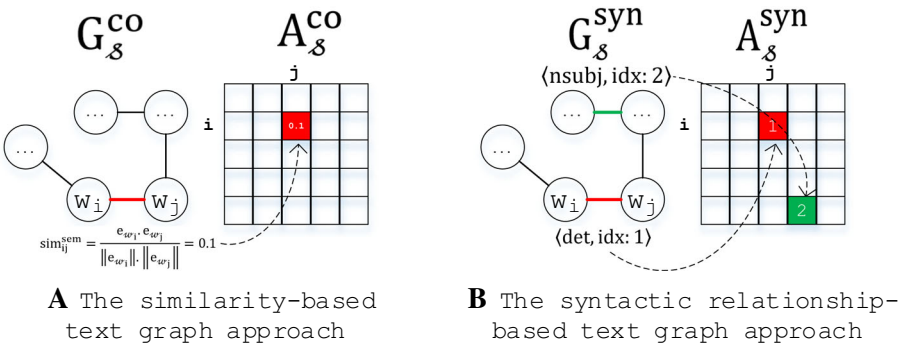
Notation	Description
$G = (V, E)$	Representing for a graph-based structure with a set of nodes (V) and edges (E)
$\mathcal{g} = \{G_1, \ldots, G_n\}$	Representing for a graph tensor
A and $\mathcal{A}$	Indicating the ad
w, $\mathfrak{z}$ and	Representing a word, a sentence, and a document, respectively
$\mathfrak{G}$ and $\mathcal{g}_i$	Representing for a set of co-referencing spans and a specific i^{th} co-referencing span
X	Denoting for an embedding matrix
e_w, $e_\mathfrak{z}$ and e	Denoting for an embedding vector of a word, a sentence and a document, respectively
Norm(.)	Representing for the normalization operation
Dropout(.)	Representing for dropout mechanism in a neural network-based architecture
$\sigma(\alpha)$	Representing for the sigmoid function, denoted as: $\sigma(\alpha) = \frac{1}{1+e^{-\alpha}}$
ReLU(α)	Representing for the Rectified Linear Units function, denoted as: $\text{ReLU}(\alpha) = \max(0,)$
softmax(α)	Representing for the softmax classification function with (K) classes, is defined as: $\text{softmax}(\alpha)_i = \frac{e^{\alpha_i}}{\sum_{j=1}^{K} e^{\alpha_j}}$

4.1 MultiTextGraphEmb: GCN-Based Graph Tensor Textual Embedding Strategy

4.1.1 Co-Occuring and Syntactic Text Graph Embedding via GCN-Based Graph Tensor

At the inital steps, we first apply a common pre-trained word embedding technique (e.g.,: Word2Vec [24], GloVe [25], fastText [27], etc.) to learn the local contextual representations of all words in each sentence. For a given sentence ($\mathfrak{z}$) with (n) in length, denoted as: $\mathfrak{z} = \{w_1, w_1, \ldots, w_n\}$, after applying word embedding technique, we achive a set of word embedding vectors as: $X_\mathfrak{z}^{\text{word\_emb}} = \{e_{w_1}, e_{w_2}, \ldots, e_{w_n}\}$, $X_\mathfrak{z}^{\text{word\_emb}} \in \mathbb{R}^{n \times d}$. Initially, to preserve the contineous latent features of words in a given sentence ($\mathfrak{z}$), we construct a word co-occurring text graph, as a graph-based structure: $G_\mathfrak{z}^{co} = (V_\mathfrak{z}^{co}, E_\mathfrak{z}^{co})$, where: $V_\mathfrak{z}^{co}$ and $E_\mathfrak{z}^{co}$ are sets of nodes and edges which represent for unique words and co-occuring relationships between two words, respectively. For evalutating the semantic relatedness between two i^{th} and j^{th} words, we apply the cosine similarity on the embedding vectors of w_i and w_j (as: e_{w_i} and e_{w_j}) to compute the semantic similarity between two given words, denoted as: $\text{sim}_{ij}^{sem} = \frac{e_{w_i} . e_{w_j}}{\|e_{w_i}\| . \|e_{w_j}\|}$. Then, the constructed co-occuring text graph of a given sentence ($\mathfrak{z}$) is a directed weighted graph, with each edge is defined as a tuple, as: $\langle w_i, w_j, \text{sim}_{ij}^{sem} \rangle \in E_\mathfrak{z}^{co}$ (as illustrated in Fig. 2a).

Then, to preserve the syntactic relationships between words, we apply the Stanford CoreNLP [17] toolkit to analyze and construct the syntactic dependency tree/graph for the given sentence ($\mathfrak{z}$), denoted as: $G_\mathfrak{z}^{syn} = (V_\mathfrak{z}^{syn}, E_\mathfrak{z}^{syn})$, with $V_\mathfrak{z}^{syn}$ is a set of nodes represent for occurring words and $E_\mathfrak{z}^{syn}$ is a set of edges represent for directed syntactic relationships between words in a given sentence ($\mathfrak{z}$). The given syntactic text graph ($G_\mathfrak{z}^{syn}$) is considered as a directed labelled graph where each edge is a data tuple, denoted as: $\langle w_i, w_j, {}_{ij}^{syc} \rangle \in E_\mathfrak{z}^{syn}$, with: ${}_{ij}^{syc}$ is the syntactical label (e.g.,: "nn", "nsubj", "det", etc. [17].) between two i^{th} and j^{th} words (as illustrated in Fig. 2b). From given constructed text graphs of a given sentence ($\mathfrak{z}$),



A The similarity-based text graph approach

B The syntactic relationship-based text graph approach

Fig. 2 Illustrations of similarity-based and syntactic dependent relationship-based text graph transformation approach in our paper

we constructed a graph tensor for sentence ($\mathcal{S}$), denoted as: $\mathcal{G}_{\mathcal{S}} = \{G_{\mathcal{S}}^{co}, G_{\mathcal{S}}^{syn}\}$ and the packed adjacency matrix of a given graph tensor is denoted as: $\mathcal{A}_{\mathcal{S}} = \{A_{\mathcal{S}}^{co}, A_{\mathcal{S}}^{syn}\}$. In more details, for the similarity-based graph, $G_{\mathcal{S}}^{co}$ which is composed by a set of 1-hop co-occurring relationships between two words, the packaged adjacency matrix $A_{\mathcal{S}}^{co}$ is formed as the similarity weights between two connected words in which each cell, as: $A_{\mathcal{S}}^{co}{}_{[i][j]}$ presents the cosine similarity weight of the corresponding w_i and w_j words. Similar to that for the packaged dependency relationship-based adjacency matrix of the given ($G_{\mathcal{S}}^{syn}$) text graph, denoted as: $A_{\mathcal{S}}^{syn}$ in which each cell, as: $A_{\mathcal{S}}^{co}{}_{[i][j]}$ presents the syntactic relationship label between two corresponding w_i and w_j words. In our approach, we mainly used the Stanford CoreNLP library to learn and extract dependency relationships between words in each sentence. For the efficiency in data pre-processing and computation steps, we index the extracted syntactic labels for relationships between words as integer numbers. For different size and complexity of evaluated text corpus, the number of extracted syntactic labels might be total different.

Since our ultimate purpose in this approach is to utilize different constructed text graphs in order to present different latent features (co-occurring and syntactic structure) of a given sentence, all constructed text graphs as heterogeneous networks which carry out varied rich-semantic meanings are packed into a single tensor graph and simultaneously learn the latent representations via the inter/intra propagation processes [16]. Let $\mathcal{H}_{\mathcal{S}}^{[t]}$ is denoted as the packed hidden state matrix at a specific time-step (t) of a given graph tensor $\mathcal{G}_{\mathcal{S}}$, which is a combined GCN-based hidden states of all text graphs in $\mathcal{G}_{\mathcal{S}}$, as: $\mathcal{H}_{\mathcal{S}}^{[t]} = \{H_{\mathcal{S}}^{[t]co}, H_{\mathcal{S}}^{[t]syn}\}$. As a GCN-based architecture, we apply the inter/intra propagation learning processes om a given graph tensor $\mathcal{G}_{\mathcal{S}}$, with a specific t^{th} hidden layer: $\mathcal{H}_{\mathcal{S}}^{[t]}$, the GCN-based graph tensor inter/intra propagation processes support to achieve the representations of nodes in the next $(t+1)^{th}$ layer (as shown in Eq. 1):

$$\begin{aligned}
&\mathcal{H}_{\mathcal{S}}^{[t+1]} = f_{inter}\left(f_{intra}\left(\mathcal{H}_{\mathcal{S}}^{[t]}\right)\right) \\
&f_{intra}\left(\mathcal{H}_{\mathcal{S}}^{[t]}\right) \rightarrow \mathcal{H}_{\mathcal{S},\alpha}^{[t]} = f_{act}\left(\widehat{\mathcal{A}}_{\mathcal{S}}.\mathcal{H}_{\mathcal{S}}^{[t]}.W_{\mathcal{S},\alpha}^{[t]}\right) \\
&f_{inter}\left(\mathcal{H}_{\mathcal{S},\alpha}^{[t]}\right) \rightarrow \mathcal{H}_{\mathcal{S}}^{[t+1]} = f_{act}\left(\widehat{\mathcal{A}}_{\mathcal{S}}.\mathcal{H}_{\mathcal{S},\alpha}^{[t]}.W_{\mathcal{S},\beta}^{[t]}\right)
\end{aligned} \tag{1}$$

where,

- $\widehat{\mathscr{A}_\flat}$, is packed normalized adjacency matrix version of a given text graph tensor, with the original packed adjacency matrix is $\mathscr{A}_\flat$.
- $W_{\flat,\alpha}$ and $W_{\flat,\beta}$, are the weighting parameter matrices of the intra propagation and inter propagation learning processes, respectively.

At the last k^{th} layer of the given GCN-based graph tensor architecture, we achieve the representations of all words as graph's nodes for each text graph in forms of packed hidden state matrix, as: $\mathscr{H}_\flat^{[k]} = \{H_\flat^{[k]^{co}}, H_\flat^{[k]^{syn}}\}$, then to obtain the final representations of all words in a given sentence ($\flat$), we apply the max-pool strategy over hidden state matrices of all text graphs, denoted as: $X_\flat^{GCN\_gt} = \text{MaxPool}\left(\mathscr{H}_\flat^{[k]}\right)$.

4.1.2 Co-Referencing Relationship Representation Learning

Next, we incorporate the co-referencing relationships between words in a given sentence ($\flat$) into the previous achieved graph tensor-based word embeddings, we apply a Bi-LSTM textual encoder with the self-attention based mechanism to inject the co-referencing latent features into the syntactical latent features of $X_\flat^{GCN\_gt}$. To do this, we first apply Stanford CoreNLP toolkit to extract co-referencing relationships between words as co-referencing spans, denoted as: $\mathfrak{G}_\flat = \{g_1, g_2, \ldots, g_{|\mathfrak{G}|}\}$. Then, we apply a Bi-LSTM architecture to capture the sequential representations of all words in each i^{th} co-referencing span (g_i) and apply the average pooling strategy over the output hidden states of the given Bi-LSTM encoder as the final representation of (g_i), denoted as: e_{g_i}. The overall processes can be formulated as the following (as shown in Eq. 2):

$$\begin{aligned}
H_{g_i}^{\text{Bi-LSTM},+\Theta} &= \text{LSTM}\left(\left\{e_{w_j}\right\}_{w_j \in g_i}, +\Theta\right) \\
H_{g_i}^{\text{Bi - LSTM},-\Theta} &= \text{LSTM}\left(\left\{e_{g_j}\right\}_{g_j \in g_i}, -\Theta\right) \\
e_{g_i} &= \text{AvgPool}\left(H_{g_i}^{\text{Bi - LSTM},+\Theta}, H_{g_i}^{\text{Bi - LSTM},-\Theta}\right)
\end{aligned} \tag{2}$$

where,

- $H^{\text{Bi-LSTM}}$, is the output hidden state of a given Bi-LSTM architecture.
- $[+\Theta, -\Theta]$, are set of model's parameters of a given Bi-LSTM architecture for the forward and backward direction, respectively.

Next, to efficiently integrate previous achieved word embedding vectors of $X_\flat^{GCN\_gt}$ into each learnt co-referencing span (g_i), we apply the self-attention mechanism over word embedding vectors in $X_\flat^{GCN\_gt}$, the defined self-attention based mechanism is formulated a full-connected neural network-based mechanism which is defined as the following (as shown in Eq. 3):

$$\begin{aligned}
Z_{g_i} &= \text{Dropout}\left(W_\beta^{sa}\right).f_{act_nl}\left(W_\alpha^{sa}.e_{g_i} + b_\alpha^{sa}\right) + b_\beta^{sa} \\
\widehat{e_{g_i}} &= \text{softmax}\left(Z_{g_i}\right) \\
e_{g_i} &\leftarrow \text{Norm}\left(\prod_{e_{g_j} \in g_i} \widehat{e_{g_i}}.e_{g_j}\right)
\end{aligned} \tag{3}$$

where,

- $f_{act\_nl}(.)$, is the non-linear activation function.
- W_{α}^{sa}, W_{β}^{sa}, b_{α}^{sa} and b_{β}^{sa}, are the weighting parameter matrices and the bias matrices of a given self-attention-based mechanism, respectively.

From the updated embedding vector (following the Eq. 3) of each co-referencing span (e_{g_i}), we apply the average pooling mechanism to update the embedding vectors of associated words in each span (g_i) in order to softly align the previous obtained syntactical structure latent features with co-referencing relatedness between words, as: $e_{w_j} \leftarrow \text{AvgPool}(e_{w_j}, e_{g_i})$ with $\bigvee g_i \in \mathfrak{G}_{\mathfrak{z}}$ and $\bigvee w_j \in g_i$. The end of this process, we will achieve the final unified embedding vectors of all words in a given sentence ($\mathfrak{z}$) which carry out the rich-semantic contextual information of both long-range dependent syntactical and co-referencing relationship structures, denoted as: $X_{\mathfrak{z}}^{MTGE}$.

4.2 BERT-Based Event-Aware Masked Language Mechanism for ED Task

In this section, we apply pre-trained BERT model with the custom event-aware masked language mechanism for promoting the ED task. Given a set of semantic-enhanced word embedding matrix, as: $X_{\mathfrak{z}}^{MTGE}$, which is obtained by the proposed MultiTextGraphEmb-based textual embedding technique, we reform the word embedding matrix as the event-aware labelled input of each sentence ($\mathfrak{z}$), for the given pre-trained BERT model as: $\mathscr{X}_{\mathfrak{z}} = \{\langle e_{w_i}, i\rangle\}_{i=1}^{n}$, where l_i is the corresponding BIO annotation label of the given i^{th} word. The purpose of using sentence-level sequential learning process of BERT over word embedding vectors is to recover the masked latent sequence of words in each sentence which are conditioned by the BIO-based labels. During the training processes of the encoder and transformer mechanisms in BERT, the training objective is to predict the word and its corresponding BIO-based label at different masked positions in each sentence ($\mathfrak{z}$). The loss function for this training objective of our approach is defined as the following (as shown in Eq. 4):

$$\mathscr{L}^{\text{BERT}} = -\sum_{\mathfrak{z}\in\mathfrak{z}} f_I(w_i).\left(\text{logProb}\left(e_{w_i}|\mathscr{X}_{\mathfrak{z}}\right) + \text{logProb}(_i|\mathscr{X}_{\mathfrak{z}})\right) \tag{4}$$

where,

- $\mathfrak{z}$, is a set of BIO-based annotated sentences in a given training set.
- $f_I(.)$, is the masked position indicator function which return value [1] if a given i^{th} word (w_i) is masked and [0] for the otherwise.

After the training process, we can obtain the word embedding vectors by using the vertical max-pooling strategy on all hidden state layers of a given k-layer BERT architecture in each sentence ($\mathfrak{z}$), as: $X_{\mathfrak{z}}^{synseq} = \text{MaxPool}\left(\{\text{BERT}^i\}_{i=1}^{k}\right)$. The final achieved word embedding matrix $X_{\mathfrak{z}}^{synseq}$ is considered as a rich-semantic contextual representations of words which jointly preserves the syntactical and the complex event-aware sequential structures of the whole given text corpus with the external rich-contextual semantic meanings of pre-trained BERT model. Then, we use this rich-semantic word embedding matrix ($X_{\mathfrak{z}}^{synseq}$) to leverage performance of multiple downstream subtasks in ED, including the multiple event detection and few-shot learning event detection.

Multiple event/argument detection. For the multiple event/argument detection task, we apply the same self-attention based trigger and argument classification mechanisms of Liu et al. [13]. In this approach, each (i^{th}) word has the contextual vector, denoted as: (e_{c_i}) which is generated by using a Bi-LSTM encoder [13] and concatenating the output hidden state vectors of both forward and backward directions. And word the self-attention score vector the

corresponding contextual embedding vector at specific (i^{th}) word is calculated and updated as the following (as shown in Eq. 5):

$$\begin{aligned} \mu_i &= \mathrm{Norm}\big(\exp\big(\mathrm{Dropout}(M_\beta)\sigma\big(M_\alpha e_{c_i} + b_\alpha\big) + b_\beta\big)\big) \\ e_{c_i} &= \left[\sum_{j=1, j\neq i}^{n} \mu_j . e_{w_j}, e_{w_i}\right] \end{aligned} \tag{5}$$

where,

- $\sigma(.)$, is the sigmoid function.
- M_α, M_β, b_α and b_β, are the corresponding weighting parameter matrices and bias matrices of the given self-attention based mechanism.

From the updated contextual embedding vector of a specific (i^{th}) word, we feed them into a full-connected MLP architecture with a softmax function at the output layer to predict the BIO-based trigger label in the given sentence ($\mathfrak{s}$), as the following (as shown in Eq. 6):

$$\begin{aligned} \gamma_i &= \mathrm{ReLU}\big(W_\alpha^{ED\_mul} . e_{c_i} + b_\alpha^{ED\_mul}\big) \\ \widehat{\ell_i} &= \mathrm{softmax}\Big(W_\beta^{ED\_mul} . \gamma_i + b_\beta^{ED\_mul}\Big) \end{aligned} \tag{6}$$

where,

- $\widehat{i}$, is the predicted BIO-based label of a specific (i^{th}) word.
- $W_\alpha^{ED\_mul}$,$W_\beta^{ED\_mul}$,$b_\alpha^{ED\_mul}$ and $b_\beta^{ED\_mul}$, are the corresponding weighting parameter matrices and bias matrices of the given event/argument classification mechanism.

To train the given multi-class event/argument detection model, we apply the stochastic gradient descent (SGD) strategy for updating model's parameters along with the corresponding calculated gradients and initial learning rate (η).

Few-shot learning event detection (FSLD). Recently, FSLD is considered as the mainstream among subtasks of ED domain due to its existing challenges as well as capability of detailing with the context of limited annotation training data. In general, FSLD is more challenging than the traditional one-event/multi-event extraction subtasks because a FSLD-based model must be able to predict unseen event types. In a few-shot learning approach, the model is trained with a given support set which contains a small set of annotated event types/classes. Then, it is trained to predict the label a query instance in accordance with the full set of event types/classes appeared in the training set. To do this, we applied the same proposed prototype encoding and FSLD-based classification mechanisms of Lai, V. D. et al. [14] to handle the FSLD task in this case. At the first stage, from a given training set, we have a support set, denoted as: (S) and the (N)-way (K)-shot setting. The given support set (S) for each query instance, denoted as: x = (q, p) is constructed as the following (as shown in Eq. 7):

$$S = \begin{bmatrix} \langle \mathfrak{s}_1^1, w_1^1, _1\rangle & \cdots & \langle \mathfrak{s}_1^K, w_1^K, _1\rangle \\ \vdots & \ddots & \vdots \\ \langle \mathfrak{s}_N^1, w_N^1, _N\rangle & \cdots & \langle \mathfrak{s}_N^K, w_N^K, _N\rangle \end{bmatrix} \tag{7}$$

where,

- $\langle 1, 2, \ldots, N\rangle$, is the set of event type labels/classes.
- $\langle \mathfrak{s}_i^j, w_i^j, _i\rangle$, is considered an instance of support set which presents for a (w_i^j)-th word in a ($\mathfrak{s}_i^j$)-th sentence is the trigger of a ($_i$)-typed event which is mentioned.

By applying the prototype encoding, we achieve the prototype representation of each (i^{th}) class/event type, denoted as: e_{et_i}, by aggregating all instances of a given class/event type, as the following (as show in Eq. 8) [14]:

$$\begin{aligned} e_{et_i} &= \sum_{\left(\vartheta_i^j, w_i^j, i\right) \in S} \psi_{ij}[e_{\vartheta_i^j}, e_{w_i^j}] \\ \text{with: } \lambda_{ij} &= \sum\left[\sigma\left(\left(\left[e_{\vartheta_i^j}, e_{w_i^j}\right]\right) \odot \left([e_q, e_p]\right)\right)\right] \\ \psi_{ij} &= \text{softmax}\left(\lambda_{ij}\right) \end{aligned} \tag{8}$$

where,

- $\sigma(\cdot)$ and $odot$, are the sigmoid function and element-wise product, respectively.
- $x = (q, p)$, presents for the query instance of a given support set (S) which has the corresponding embedding vectors as: e_q and e_p.

In previous work [14], Lai, V. D. et al. proposed a novel metric-based (e.g., cosine similarity, Euclidian distance,) with the intra/inter-cluster matching mechanisms for the few-shot learning strategy in ED, we named it as FSL-MC for short. In fact, to deal with the FSL-based event detection, we firstly apply our proposed textual embedding mechanism in this paper to handle all textul embedding tasks (instance encoder). Then, we re-implement the previous metric-based [Proto + Att] (Euclidean distance with weighted sum prototype network) with the intra/inter clustering mechanism in FSL-MC. Finally, we reapply the proposed classification module and training objectives by minimizing the negative log-likelihood of defined loss functions which are defined by Lai, V. D. et al. [14]. Finally, the trained model is then used to conduct the FSDL task in event extraction.

5 Experiments and Discussions

In this section, we present extensive experiments on benchmark datasets to demonstrate the effectiveness of our proposed SynSeq4ED model in comparing with recent state-of-the-art baselines, including: DMCNN, PSL, dbRNN, JMEE and FSL-MC. Experimental outputs show the significant performances of our proposed rich-semantic textual embedding strategy in this paper on the multiple downstream subtasks of ED area.

5.1 Dataset Descriptions and Experimental Setups

5.1.1 Descriptions of Dataset Usage

To comprehensively evaluate the performance of all event extraction models, we used two standardized datasets which are widely used in ED area: the ACE-2005 and TAC-KBP-2015. Below is the detailed information of each dataset (Table 2 describes detailed information of each dataset):

- **ACE-2005**: is considered as a classical dataset for ED domain which contains 599 documents/newswire articles where occuring words/compound words are manually annotated 33 event subtype classes and the NONE class (34 in total) and 36 role classes. This dataset can be officially achieved at this repository of LDC (Linguistic Data Consortium) [1].

[1] ACE-2005 dataset: https://catalog.ldc.upenn.edu/LDC2006T06.

Table 2 General statistics of used datasets for experiments in this paper

Dataset	No. event (sub)types/classes	Training	Testing	Validation
ACE-2005	34	529	40	30
TAC-KBP-2015	39	360	202	-

- **TAC-KBP-2015**: is a recent well-known dataset for ED area which is officially introduced and used in the Event Nugget Detection Evaluation of the 2015 Text Analysis Conference (TAC). This dataset contains 562 documents (360 for training and 202 for testing) which are annotated by 38 event subtype classes + 1 NONE class (39 in total) for the ED task. This dataset can be achieved from the official website of TAC at:[2].

Dataset pre-processing steps. We mainly apply the Stanford CoreNLP library [17] for textual pre-processing steps, includes: word tokenizing, sentence splitting, POS analysis & tagging, dependency and co-reference relationship extractions.

5.1.2 Evaluation Metric and Configurations

Similar to previous studies [12, 13, 14], to evaluate the accuracy performance of all ED-based models, we mainly used the F1 accuracy metric for scoring the event classification output results. In general, the F1 metric considers both precision (P) and recall (R) values of clustering outputs to compute the F1 value. The F1 metric is used for evaluating the event detection task can be formally defined as the following (see Eqs. 9 and 10):

$$P = \frac{TP}{TP + FP}, \quad R = \frac{TP}{TP + FN} \tag{9}$$

$$F1 = 2.\frac{P.R}{P + R} \tag{10}$$

where:

- TP, is the number of correct entities which are assigned to their correct labels.
- FP and FN, are number of expected entities which are assigned to specific labels but not correct and not assigned by actually belong to these labels, respectively.

In addition To initally learn the local contextual representations of words in each datasets, we used the 300-dimension pre-trained Word2Vec model[3]. For the pre-trained BERT textual encoder, we reused the official large/uncased version from Google which can be achieved at this repository[4]. Specifically, for our proposed MultiTextGraphEmb-based embedding approach, we set the number of layers for the given GCN architecture and the number of LSTM-based cells for the given Bi-LSTM architecture to 10 and 164, respectively. The full model's parameter configurations are listed in Table 3.

[2] TAC-KBP-2015 dataset: https://tac.nist.gov/2015/KBP/data.html.

[3] Pre-trained Word2Vec (300-dimension): https://code.google.com/archive/p/word2vec/.

[4] Pre-trained BERT (large, uncased): https://github.com/google-research/bert.

Table 3 Experimental configurations for our proposed SynSeq4ED model

Model's parameter	Value
Word embedding vector dimensional size (d_{word_emb})	300
GCN-based node feature embedding vector dimensional size (d_{GCN})	300
Number of hidden LSTM-based cells for Bi-LSTM architectures	164
Number of layers for GCN architectures (k_{GCN})	10
Default learning rate value (η)	0.0001
Default number of training epochs	450

5.1.3 Experimental Comparative Baselines

For comparing the accuracy performances of our proposed SynSeq4ED model with recent baselines, we implemented several well-known ED-based models for both one/multiple and few-shot learning event detection tasks, which are:

- **DMCNN** [10]: is a CNN-based model with the custom multi-pooling strategy for textual representation learning process which enables to facilitate the process of preserving the structural information event/argument. The proposed DMCNN model has demonstrated efficiencies in automatically lexical-level and sentence-level latent feature extraction from texts for ED without much supports from complicated NLP toolkits.
- **PSL** [20]: is a probabilistic reasoning based model for ED task which is also considered as a text embedding-based technique. The proposed PSL is designed upon the Probabilistic Soft Logic approach which supports to jointly encode the latent feature and global information of texts for effectively handling ED task. Experimental results in ACE dataset demonstrates the effectiveness of PSL in multiple downstream tasks of event extraction.
- **dbRNN** [21]: is a recent well-known RNN-based approach for ED which utilizes the Bi-LSTM-based sequential encoder with custom bridge dependency mechanism for leveraging the performance of ED task. The proposed dbRNN model supports to fully preserve the syntactic and sequential representations via the semantic-enhanced syntactical dependency analysis during the process of modeling occurring words in texts. Through experiments, dbRNN presents significant performances in event/argument classification task.
- **JMEE** [13]: is a recent integrated GCN-based syntactic encoding with attention-based mechanism for efficiently learning the representations of texts which supports to improve the performance of event extraction. In JMEE model, Liu, X. proposed a self-attention mechanism to aggregate the relationships and associated information between events/arguments for dealing multiple event trigger/argument classification tasks. With advantages of capability for handling multi-class event/argument classification, the JMEE model is considered as our main competitor in this paper.
- **FSL-MC** [14]: is considered as the earliest effort in applying the few-short learning strategy for the event/argument extraction from texts. In FSL-MC model, Lai, V. D. et al. proposed a novel metric-based cluster matching with prototype encoding mechanisms to efficiently model the representation of event types in support sets over query instances for FSL-based ED task. Extensive experiments in ACE benchmark dataset demonstrates the great achievements of Lai, V. D. et al. in FSL-based ED area. Majorly inspired from the FSL-MC

model for FSL-based event extraction, we considered FSL-MC as our main competitor for experimental comparisons in FSL-based ED task.

For most of general configurations which are configured for these comparative baselines, we used the same of our proposed SynSeq4ED model (described in Table 3). For the other specified setting parameters of each model, we used the same configurations in the original works which achieve the best accuracy performance in ED task.

5.2 Experimental Outputs and Discussions

5.2.1 Multiple Event Detection Task

Similar to previous empirical studies in [10, 13], we implemented each model to conduct different subtasks of ED, including: event trigger identification/classification and argument identification/classification in the context of multi-class classification problem. For each model, we repeated experiments 10 time and reported the average outputs in terms of F1 score as the final result. Tables 4 and 5 show the experimental outputs for multiple ED-based subtasks by using different models in ACE-2005 and TAC-KBP-2015 datasets.

In general, the experimental outputs show that our proposed SynSeq4ED model achieved the highest accuracy performances in terms of F1 score for all subtasks, includes: event trigger identification/classification, argument identification and argument role classification in comparing with recent state-of-the-art ED-based methods. In more details, our proposed SynSeq4ED model outperforms the DMCNN, PSL and dbRNN averagely about 25.37%, 3.85% and 4.48% for overall ED's subtasks in both two ACE-2005 and TAC-KBP-2015 datasets. For our main competitor in multiple event extraction task, the JMEE model our proposed SynSeq4ED model also slightly achieves higher accuracy performance approximately 3.06% for all 4 subtasks in two given benchmark datasets. To further evaluate the stability of our proposed model for event classification task with different size of training data, we varied the size of training set (from 10 to 100%) of ACE-2005 and TAC-KBP-2015 datasets and conducted again the experiments for the event trigger classification subtask. For different training set sizes, we reported the changes of accuracy performance of our proposed SynSeq4ED and JMEE model. The experimental outputs as shown in Fig. 3 presents that our proposed SynSeq4ED performs well on the event classification task in comparing with JMEE model with less training size (> 80%) to achieve the highest accuracy performance in terms of F1 metric. These extra empirical studies also indicate that our proposed SynSeq4ED model can be quite stable for handling ED-based subtasks with a small size of training data.

Table 4 Experimental outputs for multiple subtasks of ED via different model in the ACE-2005 dataset

Model/Subtasks	Event trigger identification	Event trigger classification	Argument identification	Argument role
DMCNN	72.236	61.293	52.239	49.682
PSL	–	69.678	–	–
dbRNN	–	72.821	68.671	66.802
JMEE	76.921	74.281	70.213	62.678
SynSeq4ED	**78.791**	**77.291**	**71.692**	**64.782**

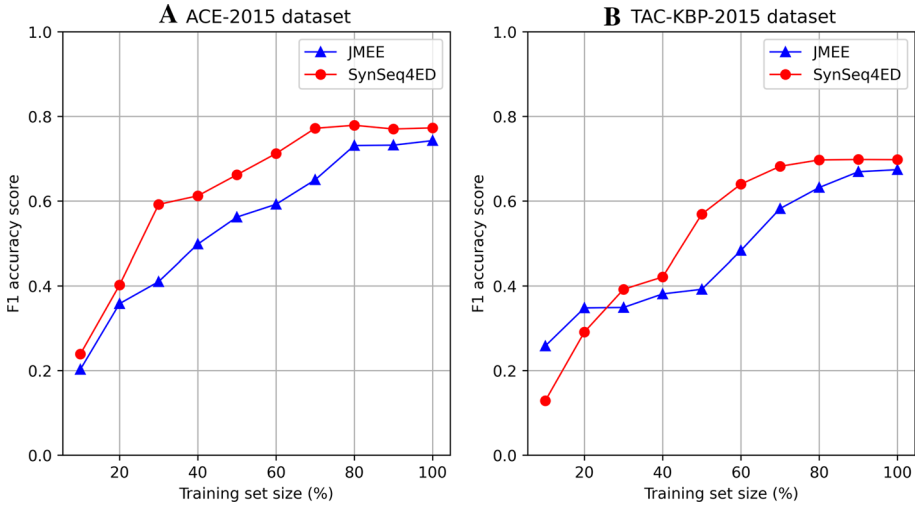


Fig. 3 Accuracy performances of JMEE and our proposed SynSeq4ED models with different size of training set (%) of ACE-2005 and TAC-KBP-2015 datasets

5.2.2 Few-Shot Learning Event Detection Task

For the few-short learning event extraction task which is considered as a new approach in the ED area, therefore we only compared the performance of our proposed SynSeq4ED model with the primitive FSL-MC baseline of Lai et al. [14]. All experimental configurations related to the setups of ACE-2005 dataset, training/testing/validation splitting and model's hyper-parameters are setup the same as the original work [14]. We applied the [Proto + Att] metric-based approach for both FSL-MC and SynSeq4ED models and reported the average accuracy performance of FSL-based ED task in terms of F1 metric.

The experimental outputs for FSL-based [5 + 1-way 5-shot] and [10 + 1-way 10-shot] settings in Table 6 demonstrates that our proposed event-aware BERT-based textual embedding can support to achieve better performance than the previous proposed instance encoding mechanism of FSL-MC model. The experimental outputs in this section prove that our proposed textual embedding mechanism can be flexible applied in FSL-based ED task. The given textual embedding strategy in our proposed SynSeq4ED model is a combination between the GCN and pre-trained BERT architectures to effectively capture the long-range dependent syntactic and sequential representations of words which is considered as a suitable substitution for text embedding approach in previous ED-based models.

5.2.3 Experimental Studies on Parameter Sensitivity

Taking advances of the recent deep learning architectures, such as: Bi-LSTM, GCN and deep textual embedding mechanisms (e.g., Word2Vec, BERT, etc.), our proposed SynSeq4ED model might be sensitive with setup deep learning based parameters such as embedding vector dimensionality, number of LSTM-based cells usage, etc. which can directly affect the overall performance of ED-based tasks. Therefore, in this section, we demonstrated extensive experiments related to the sensitivity of important configuration parameters in our proposed SynSeq4ED model. First of all, we evaluate the influence of the dimensionality of learnt word

Table 5 Experimental outputs for multiple subtasks of ED via different model in the TAC-KBP-2015 dataset

Model/Subtasks	Event trigger identification	Event trigger classification	Argument identification	Argument role
DMCNN	59.281	51.921	48.418	47.291
PSL	–	63.682	–	–
dbRNN	–	68.281	62.868	58.283
JMEE	62.239	67.421	64.219	59.881
SynSeq4ED	**64.682**	**69.781**	**65.441**	**61.829**

Table 6 Experimental outputs of FSL-based task with different models in ACE-2005 dataset

	5 + 1-way 5-shot	10 + 1-way 10-shot
JMEE	79.291	73.689
FSL-MC	**81.682**	**75.078**

embedding vectors (d) (which is also the dimensionality of node feature vectors in the given GCN-based archiectures) on the overall accuracy performance of our proposed SynSeq4ED model. To do this, we varied the value of (d) parameter from 10 to 400 and reported the changes of accuracy performance for the event classification subtask in the ACE-2005 and TAC-KBP-2015 datasets. Experimental outputs (as shown in Fig. 4) demonstrate that our proposed SynSeq4ED model is quite sensitive with the (d) parameter where a suitable value of (d) must be selected upon the extensive considerations of dataset related aspects (e.g., size, complexity, etc.). In more details, for a larger and complex dataset like as ACE-2005, it

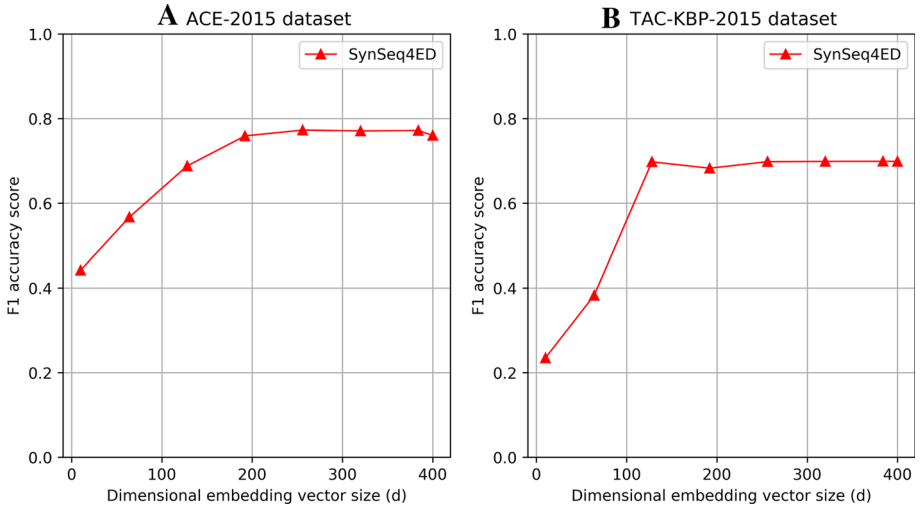


Fig. 4 The influence of dimensionality of word embedding vector, (d) parameter on overall SynSeq4ED model accuracy for the event trigger classification subtask

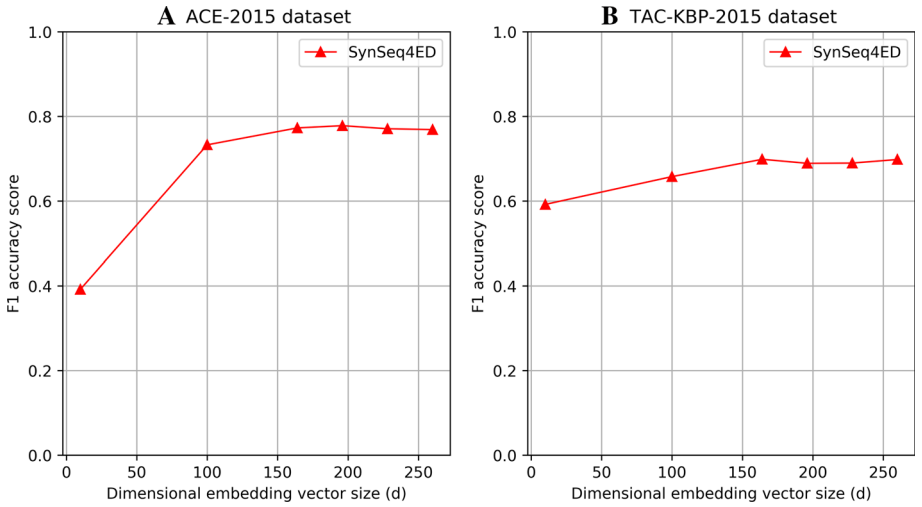


Fig. 5 The influence of number of used LSTM-based cells for Bi-LSTM encoder, (h) parameter on overall SynSeq4ED model accuracy for the event trigger classification subtask

needs about > 256 in value of (d) parameter to make our model reach the highest performance, whereas it needs only about > 128 for the TAC-KBP-2015 dataset.

Next, we study the influence of number of used LSTM-based cells, as: (h) for Bi-LSTM encoders which are used in our proposed SynSeq4ED model. Similar to experiments related to the sentivity of (d) parameter, we also varied the value of (h) parameter from 10 to 260 and reported the fluctuations of overall model's accuracy performances for the event classification tasks in all datasets. As shown from the experimental outputs (in Fig. 5), our proposed model is insensitive with the (h) parameter where the highest performances in all datasets are reached at the value of (h) parameter > 164.

5.2.4 Studies on Different Word Embedding Techiniques

In this section, we present extensive studies on the influences of different word embedding techniques which are used in our proposed SynSeq4ED model. To do this, we implemented our word embedding mechanism by using different well-known word representation learning techniques which are: Word2Vec [24], GloVe [25] and fastText [27]. For three different versions of SynSeq4ED model (as: SynSeq4ED-Word2Vec, SynSeq4ED-Glove and SynSeq4ED-fastText), we used them to handle the event detection task in ACE-2005 and TAC-KBP-2015 datasets. The performance of each word embedding techniques are evaluated under the F-1 accuracy metric with different values of embedding vector dimensionality.

Figure 6 shows the comparative studies on using different word embedding techniques in our proposed SynSeq4ED model. As shown from the experimental outputs, all word embedding techniques nearly gain the same accuracy performances for the event detection task. However, in the TAC-KBP-2015 (as shown in Fig. 6b), the classical Word2Vec [24] method slightly outperforms the GloVe and fastText at the stability aspect. In more details The SynSeq4ED-Word2Vec version rapidly achieves the highest accuracy performance with the dimensionality of embedding vector is over 130 and is stably kept at this convergent point for different higher values of embedding vector size. In contrast, all the SynSeq4ED-GloVe and

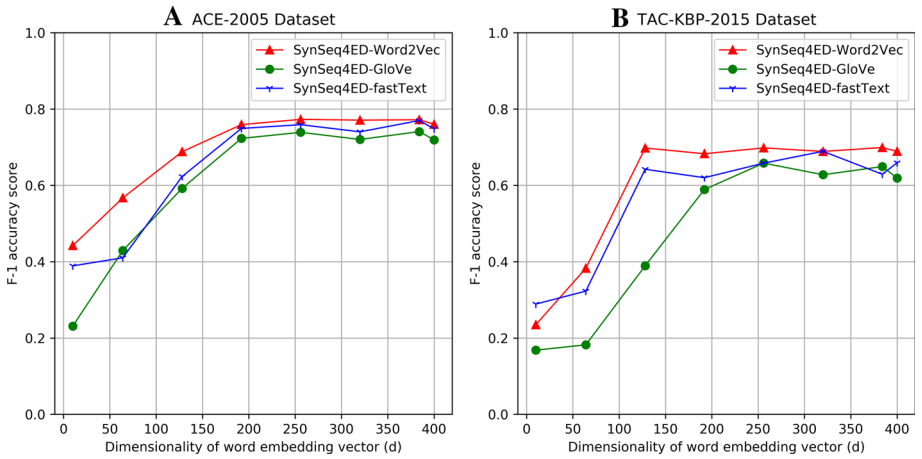


Fig. 6 Comparative studies between different word embedding methods which are implemented in our proposed SynSeq4ED for ED task in ACE-2005 and TAC-KBP-2015 datasets

SynSeq4ED-fastText versions suffer the oscillations with different values of word embedding vector size. Thus, through these extensive experiments, the Word2Vec embedding technique can be assumed as the ideal word embedding mechanism for our approach in this paper.

5.2.5 Impacts of Semantic Text Graph Representation Learning and Pre-Trained BERT Model

Effectiveness of semantic text graph representation learning In this section, we present ablation studies related the influences of applying semantic text graph representation learning and pre-trained BERT model in our proposed SynSeq4ED model. First of all, to study the impacts of applying similarity and syntactic dependency graph representation learning in our proposed model, we implemented separated versions of SynSeq4ED, which are: SynSeq4ED-SimGraph (only uses similarity-based graph transformation), SynSeq4ED-SynGraph (only uses syntactic dependency-based graph transformation) and full-featured SynSeq4ED model. We tested the performance of these three versions with the ED task in different sizes (%) of the ACE-2005 and TAC-KBP-2015 datasets.

Experimental outputs in Fig. 7 show the usefulness of applying similarity and syntactic dependency graph representation learning in our proposed SynSeq4ED model. As shown from experiments, the combined use of similarity-based and syntactic dependency-based graph representation learning techniques in our proposed MultiTextGraphEmb-based textual embedding mechanism can help to significantly improve the performance of ED task in both ACE-2005 and TAC-KBP-2015 datasets. Moreover, as shown from the experiments in Fig. 7, the separated uses of similarity-based (blue line) and syntactic dependency-based (green line) transformations achieve remarkably lower accuracy performance than the combined version.

Pre-trained BERT vs. Bi-LSTM in sequential textual representation learning for ED task In addition, to further evaluate the influence of using different sequential textual embedding methods (BERT model is used in our works), we conduct an extensive comparison between the use of BERT and Bi-LSTM (similar to JMEE [13] model) sequential encoder in our proposed SynSeq4ED. Two versions of SynSeq4ED (SynSeq4ED-BERT and SynSeq4ED-Bi-LSTM) are implemented to handle ED task in different sizes of ACE-2005

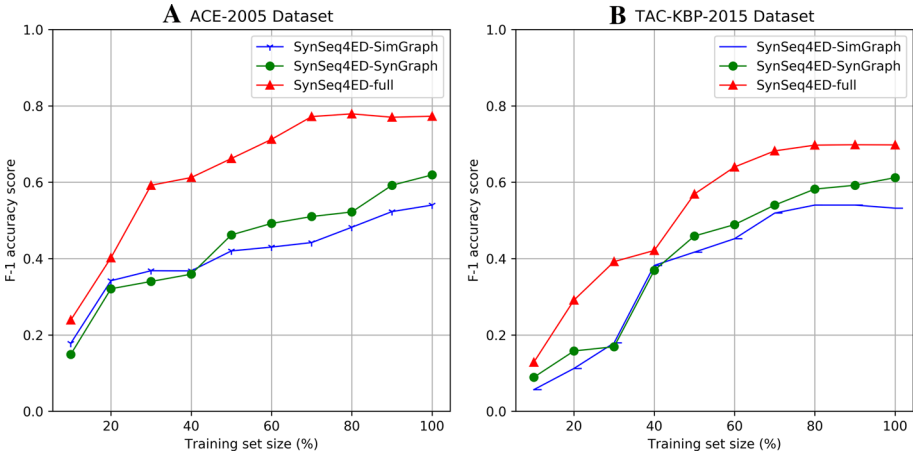


Fig. 7 Evaluations on the effectiveness of similarity-based and syntactic-based text graph transformation and learning for ED task in ACE-2005 and TAC-KBP-2015 datasets

and TAC-KBP-2015 datasets. As shown from the experimental output in Fig. 8, the integrated BERT-based version of our SynSeq4ED model remarkably gains better performance than the Bi-LSTM-based version, averagely 14.61% in both ACE-2005 and TAC-KBP-2015 datasets. This extensive experiment proves the fact that applying pre-trained BERT with linguistic masking mechanism can significantly support to leverage the performance of ED task.

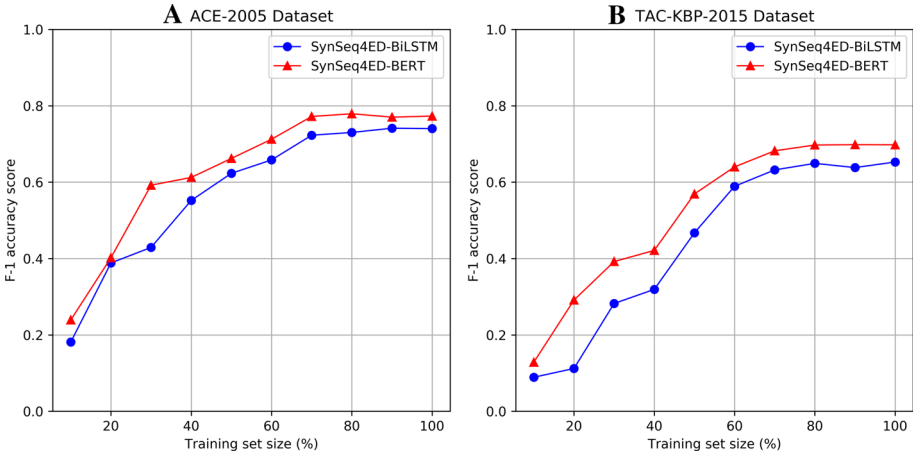


Fig. 8 Comparative studies on the performance of BERT and Bi-LSTM for sequential textual representation learning techniques in our proposed SynSeq4ED model

6 Conclusions and Future Works

In conclusion, our works in this paper are mainly concentrated on a novel proposal of deep neural textual embedding technique for ED task, called as SynSeq4ED. In our proposed SynSeq4ED model, we formally introduce a novel syntactical and co-referencing relationship based latent structural representation learning via GCN-based graph tensor encoding technique, named as MultiTextGraphEmb. Our proposed graph tensor based textual embedding technique can effectively support to jointly capture the long-range syntactical dependency and co-referencing relationships between words in texts. Then, the achieved word representations by MultiTextGraphEmb-based strategy are used to feed into a pre-trained BERT model with the event-aware masked language mechanism to leverage the performance of multiple subtasks in ED area, such as: multiple event/argument detection and few-shot learning event detection. Extensive experiments in benchmark datasets demonstrate the effectiveness of our proposed SynSeq4ED model in comparing with recent state-of-the-art ED-based techniques. In our future works, we intend to expand and apply our proposed MultiTextGraphEmb textual embedding strategy into other areas of NLP such as sentiment analysis and text-stream clustering.

Acknowledgements This research is funded by Thu Dau Mot University, Binh Duong, Vietnam.

Funding This study was funded by Thu Dau Mot University, Binh Duong, Vietnam.

References

1. Parcheta Z, Sanchis-Trilles G, Casacuberta F, Rendahl R (2020) Combining embeddings of input data for text classification. Neural Process Lett 1–29
2. Label-embedding bi-directional attentive model for multi-label text classification. Neural Process Lett 53(1): 375–389 (2021)
3. Venugopal D, Chen C, Gogate V, Ng V (2014) Relieving the computational bottleneck: Joint inference for event extraction with high-dimensional features. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP).
4. Yang B, Mitchell T (2016) Joint extraction of events and entities within a document context. In Proceedings of NAACL-HLT
5. Ding C, Hu Z, Karmoshi S, Zhu M (2017) A novel two-stage learning pipeline for deep neural networks. Neural Process Lett 46(1):159–169
6. Nguyen TH, Cho K, Grishman R (2016) Joint event extraction via recurrent neural networks. In In Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies
7. Ghaeini R, Fern X, Huang L, Tadepalli P (2016) Event nugget detection with forward-backward recurrent neural networks. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics
8. Chen Y, Liu S, He S, Liu K, Zhao J (2016) Event extraction via bidirectional long short-term memory tensor neural networks. In Chinese Computational Linguistics and Natural Language Processing Based on Naturally Annotated Big Data
9. Nguyen TH, Grishman R (2015) Event detection and domain adaptation with 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
10. Chen Y, Xu L, Liu K, Zeng D, Zhao J (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
11. Nguyen TH, Grishman R (2016) Modeling skip-grams for event detection with convolutional neural networks. In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing

12. Nguyen T, Grishman R (2018) Graph convolutional networks with argument-aware pooling for event detection. In: Proceedings of the AAAI Conference on Artificial Intelligence
13. Liu X, Luo Z, Huang HY (2018) Jointly Multiple Events Extraction via Attention-based Graph Information Aggregation. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing
14. Lai VD, Nguyen TH, Dernoncourt F (2020) Extensively matching for few-shot learning event detection. In: Proceedings of the First Joint Workshop on Narrative Understanding, Storylines, and Events
15. Devlin J, Chang MW, Lee K, Toutanova K (2019) Bert: Pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies
16. Liu X, You X, Zhang X, Wu J, Lv P (2020) Tensor graph convolutional networks for text classification. In: Proceedings of the AAAI Conference on Artificial Intelligence
17. Manning CD, Surdeanu M, Bauer J, Finkel JR, Bethard S, McClosky D (2014) The Stanford CoreNLP natural language processing toolkit. In: Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations
18. Xiang W, Wang B (2019) A survey of event extraction from text. IEEE Access 7:173111–173137
19. Li J, Luong MT, Jurafsky D, Hovy E (2015) When are tree structures necessary for deep learning of representations? In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing
20. Liu S, Liu K, He S, Zhao J (2016) A probabilistic soft logic based approach to exploiting latent and global information in event classification. In: Proceedings of the AAAI Conference on Artificial Intelligence.
21. Sha L, Qian F, Chang B, Sui Z (2018) Jointly extracting event triggers and arguments by dependency-bridge RNN and tensor-based argument interaction. In: Proceedings of the AAAI Conference on Artificial Intelligence
22. Zhang W, Ding X, Liu T (2018) Learning target-dependent sentence representations for chinese event detection. In: China Conference on Information Retrieval
23. Tai KS, Socher R, Manning CD (2015) Improved semantic representations from tree-structured long short-term memory networks. In: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing,.
24. Mikolov T, Chen K, Corrado G, Dean J (2013) Efficient estimation of word representations in vector space. In: 1st International Conference on Learning Representations (ICRL)
25. Pennington J, Socher R, Manning CD (2014) Glove: Global vectors for word representation. In: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)
26. Kipf TN, Welling M (2017) Semi-supervised classification with graph convolutional networks. In: 5th International Conference on Learning Representations, ICLR
27. Mikolov T, Grave É, Bojanowski P, Puhrsch C, Joulin A (2018) Advances in pre-training distributed word representations. In: Proceedings of the Eleventh International Conference on Language Resources and Evaluation