

Syntax-aware Event Temporal Relation Extraction Using Constraint Graph

Haijiao Liu, Jie Zhou, Xin Zhou, Fei Hu, Jiaqian Yin and Xiaodong Wang[✉]

College of Computer, National University of Defense Technology,
Changsha 410073, Hunan, China
xdwang@nudt.edu.cn

Abstract. Extracting temporal relations among events is essential in natural language understanding tasks. When two event mentions are widely separated in one text, the contextual information between them often becomes complicated and the temporal clues are difficult to locate, making inferring their temporal relationship more challenging. In this paper, we propose a novel approach named **Constraint Graph-based and Syntax-aware Event Temporal Relation Extraction (CGSE)** to address this issue. Specifically, we build temporal constraint rules by event attributes from databases to obtain prior temporal knowledge. Then we construct constraint graphs based on temporal constraint rules and present a graph neural network to model the dependencies. To eliminate irrelevant information in complicated contexts, we employ the Shortest Dependency Paths (SDP) between events in syntactic dependency parse trees, while also incorporating more temporal clues into the SDP. After that, we utilize a graph transformer to learn the representation of the SDP. Finally, a constraint fusion module is used to integrate constraint information and syntactic information to improve performance further. Experiments on two benchmark datasets, MATRES and TB-DENSE, establish that our proposed method demonstrates remarkable superiority over the previously existing state-of-the-art approaches in temporal relation extraction.

Keywords: Event temporal relation, Constraint graph, Dependency parse trees

1 Introduction

Event temporal relation extraction (ETRE) aims to infer the temporal order between two events from text and plays a crucial role in various downstream applications, such as summary [1, 2], question answering [3] and story generation [4], etc. Recent studies have achieved significant performance improvements by neural network-based methods [5–13] and pre-trained language models [14–25].

However, few previous works have analyzed and addressed the influence of the distance between two events in a text on the ETRE task. When two events are far apart, the number of words between them increases, causing the contextual information to become more complex, such as $\langle e1, e4 \rangle$ in Fig. 1. Many temporal relationship clues are hidden in the contextual information between and around events [20]. Thus, one of the persistent challenges in ETRE is the difficulty in locating temporal clues relevant

to the events due to the increased complexity of contextual information, particularly in long-distance event pairs. For example, we observe a decrease in the performance of pre-trained language models like RoBERTa [26] as the number of tokens between events increases. To fill this research gap, we aim to develop a novel method that obtains additional prior temporal knowledge and effectively captures temporal clues in long-distance event pairs.

Specifically, we have discovered that event trigger words can obtain their tenses and event types to get more effective prior temporal information. For example, the event type of e_1 is Occurrence and the event type of e_3 is *Reporting*, both of them are in the past tense. Since the corpus is sourced from news, *Reporting* always happens after *Occurrence*, like e_1 and e_3 . Hence, we define $(/Occurrence/Past, Before, /Reporting/Past)$ as a temporal constraint rule. The event attributes, including the event tense and type, have already been annotated in the datasets. Furthermore, the datasets contain more than just the mentioned temporal constraint rule. Given the tense and type of events, we can build additional temporal constraint rules, which offer more external event information and temporal clues. Therefore, temporal constraint rules serve as an external knowledge base and provide basic and general knowledge.

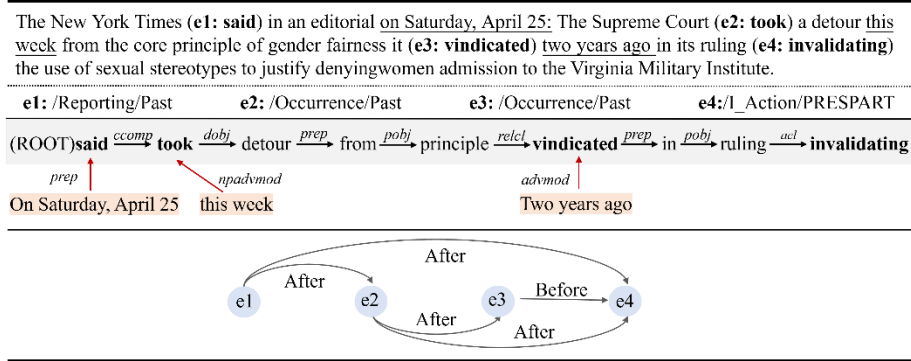


Fig. 1. Example of ETRE. Event mentions are highlighted in bold. Below event attributes and the SDP are provided with explicit temporal clues marked in orange. The following graph depicts the temporal relationships of event pairs.

Since the event pairs have different semantic relationships in diverse contexts, it is crucial to analyze the contextual information associated with event pairs to infer their temporal order. Previous works [5, 6, 14, 20] have shown the shortest dependency path (SDP) in syntactic dependency parse trees can effectively filter out irrelevant information in complex contexts and retain the most relevant words of events by focusing on the action and agents in sentences. Since the SDP is to extract the syntactic structure for two event mentions in sentences, some modification information will be removed. But in some cases, useful explicit time clues exist in this modification information, as exemplified in $\langle e_2, e_3 \rangle$ in Fig. 1. "this week" and "two years ago" represent the temporal information of the two events, but they are not included in the SDP. To tackle this issue, we incorporate more neighbor information around event mentions, such as time

words(e.g., yesterday, now), tense words(e.g., was, have), and discourse punctuation(e.g.,:, " ") into SDP to locate more explicit temporal information.

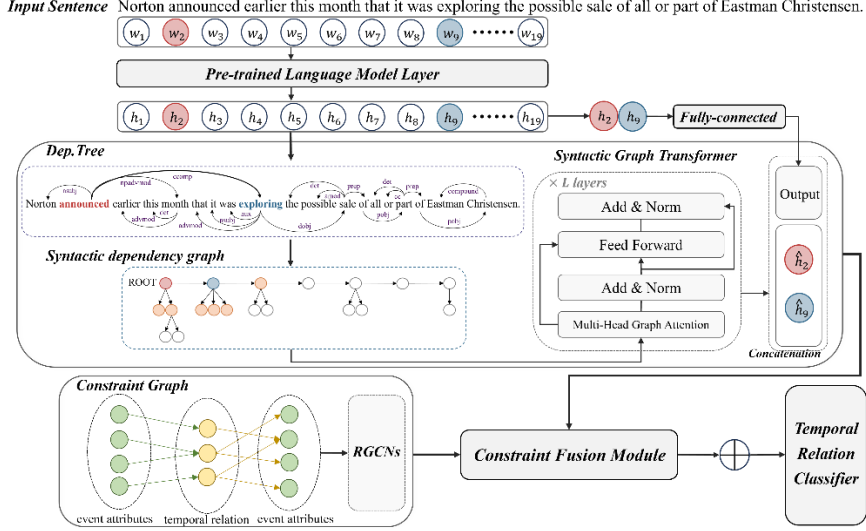


Fig. 2. Architecture overview. The event tokens are highlighted in red and blue. In the syntactic dependency graph, the orange nodes represent the neighbor nodes of event pairs.

Based on the above analysis, we propose a novel approach called Constraint Graph-based and Syntactic-guided Event Temporal Relation Extraction (CGSE). To enable our model to locate more temporal clues and eliminate irrelevant information within the complex context, we adopt the following strategies: Firstly, we build temporal constraint rules based on event attributes from databases. In particular, when mapping temporal constraint rules onto a directed graph, we represent the temporal relation label as a node instead of an edge attribute to uncover its deeper underlying meaning. In other words, each temporal constraint rule is represented by two directed edges on the constraint graph, where each edge connects a temporal relation label node and a temporal attribute node, and vice versa. Then, we employ Relation Graph Convolution Networks (RGCNs) [27] as the graph encoder to extract interactive information from the constraint graph.

Secondly, we utilize a public dependency parser to obtain the SDP and incorporate more information into it. Different from most approaches that use graph neural networks to encode the SDP, we utilize a graph transformer to capture long distance dependencies better and reduce excessive smoothing. Finally, our constraint fusion module integrates constraint information and syntactic information in a soft way to compute the final representation, improving performance in predicting temporal relations between events. We conduct experiments on two renowned benchmark datasets, MATRES [28] and TB-Dense [29] to demonstrate the superior effectiveness of CGSE over previous state-of-the-art methods and excellent performance.

2 Method

Following previous works, we formulate the ETRE problem as a multi-class classification task. Given an event pair $\langle e_i, e_j \rangle$ our goal is to predict the temporal relation label between them with sentences they belong to. Fig. 2 shows the overview of our approach. Our work CGSE involves four major parts: (i) Constraint Graph Construction to build temporal constraint rules and graphs based on event attributes from databases. (ii) Constraint Graph Encoder to obtain the representations of temporal relation labels and event attributes. (iii) Syntactic Graph Transformer to transform words into embedding vectors and extract the syntactic structure of corresponding sentences. (iv) Constraint Fusion Module to combine outputs from the two above modules and then compute the complete representation to make the final prediction.

2.1 Notations and Definitions

Given a constraint graph $\mathcal{G}_c = \{\mathcal{T}, \mathcal{R}_t, \mathcal{U}, \mathcal{C}\}$ with $\mathcal{T} = \mathcal{T}_p * \mathcal{T}_e$ and $\mathcal{U} = \mathcal{T} \cup \mathcal{R}_t$, where $\mathcal{T}_p, \mathcal{T}_e, \mathcal{U}, \mathcal{R}_t, \mathcal{C}$ indicate the sets of event types, event tense, constraint nodes, temporal relation labels and constraints, respectively. Each constraint $(t_{e_i}, r_{t_{ij}}, t_{e_j}) \in \mathcal{C}$ indicates a constraint rule. We define a syntactic graph as $\mathcal{G}_s = \{\mathcal{N}, \mathcal{R}_d, \mathcal{P}\}$, where $\mathcal{N}, \mathcal{R}_d, \mathcal{P}$ represent the sets of token nodes, dependency relations and dependency path, respectively. In SDP, each triple $(n_i, r_{d_{ij}}, n_j) \in \mathcal{P}$ indicates syntactic dependency relation $r_{d_{ij}}$ between two tokens n_i, n_j in sentences.

2.2 Constraint Graph Construction

Constraint graphs provide more basic and general knowledge, direct rule constraints, and a compact vector space for temporal knowledge mining. According to TimeML guidelines [30], the event type is categorized into seven types: *Occurrence*, *Reporting*, *Perception*, *Aspectual*, *I_Action*, *I_State* and *State*. Considering that most events are expressed as verbs, their tense can be classified into four categories: *Past*, *Present*, *Future* and *None*. The constraint rules are constructed by the temporal information found in both event tense and event type.

Table 1. Event Type Constraint

	Occurrence	Reporting	I_Action	I_State	State	Aspectual	Perception
Occurrence	-	Before	Vague	Vague	Vague	Vague	Vague
Reporting	After	-	After	After	After	Vague	After
I_Action	Vague	Before	-	Vague	Vague	Vague	-
I_State	Vague	Before	Vague	-	After	Vague	-
State	Vague	Before	Vague	Before	-	Vague	-
Aspectual	Vague	Vague	Vague	Vague	Vague	-	-
Perception	Vague	Before	-	-	-	-	-

To ensure high-quality constraint rules, we prioritize filtering based on the inherent temporal relationships of event tenses. In cases where the event tenses are consistent or "None", the implicit temporal relationships between event types are used to filter further, see Table 1. Event type constraint can be constructed according to the co-occurrence frequency of each temporal relation with event type pairs in the dataset. As shown in Appendix A, for example, (*Occurrence*, *Before*, *Reporting*) and (*Reporting*, *After*, *Occurrence*) sample numbers are 4 to 36 times greater than other temporal relationships respectively. The temporal relation itself exhibits symmetry (i.e., *Before* and *After*, *Includes* and *Is_Included*), reflexivity (i.e., *Vague*, *Equal*), and transitivity. Thus, we can establish two constraints for event types. If the number of samples between two event types is uniformly distributed over the temporal labels, its relation is defined as "Vague". In addition, if the number is too small or even 0, it indicates no clear temporal constraint between the two types and is represented as '-' in Table 1. We filter a total of 1051 constraint rules, and then apply the reflexivity of temporal relationships (e.g., *before* and *after*), resulting in a final count of 375 constraint rules. Note that one constraint graph is shared by all the inputs from the same database.

2.3 Constraint Graph Encoder

We first obtain constraint graphs and vector representations from the input layer. Next, we apply RGCNs [27] to extract the interactive features of the nodes and acquire node representations. Finally, the representations of event attributes and temporal relation labels can be derived by dividing the node representations.

Input Layer For each constraint $(t_{e_i}, r_{t_{ij}}, t_{e_j}) \in \mathcal{C}$ with $t_{e_i}, t_{e_j} \in \mathcal{T}$ and $r_{t_{ij}} \in \mathcal{R}_t$, we add $(t_{e_i}, r_{t_{ij}})$ and $(r_{t_{ij}}, t_{e_j})$ into the edge set \mathcal{E} to obtain a constraint graph. With the edge set \mathcal{E} , the adjacency matrix $\hat{\mathbf{A}} \in R^{m \times m}$ ($m = |\mathcal{U}|$) is defined as:

$$\hat{\mathbf{A}}_{ij} = \begin{cases} 1 & \text{if } (u_i, u_j) \in \mathcal{E} \text{ } (u_i, u_j \in \mathcal{U}), \\ 0 & \text{otherwise.} \end{cases} \quad (1)$$

For each node $u_i \in \mathcal{U}$, we randomly initialize a d_u dimensional embedding $\mathbf{u}_i^{(0)}$. By following the above process, the raw constraint graph is transformed into an embedding matrix $\mathbf{U}^{(0)} = \{\mathbf{u}_1^{(0)}, \mathbf{u}_2^{(0)}, \dots, \mathbf{u}_m^{(0)}\}$ with an adjacency matrix $\hat{\mathbf{A}}$.

Encoding Layer In this study, we employ RGCNs to encode the constraint graph information. RGCNs are primarily motivated as an extension of GCNs [31] that operate on local graph neighborhoods to relational data. Similar to GCNs, RGCNs can effectively promote information propagation in the graph with the neighborhood integration mechanism, which effectively considers the various edge types presented in the graph in the message-passing mechanism. Nevertheless, RGCNs extend the single graph processing of GCNs to handle multiple graphs and expand the node classification in GCNs to graph classification.

The nodes in the constraint graph are composed of event nodes and time relation nodes, where $r \in R_t$ is a relation type. In this case, the relation specific adjacency

matrix $\widehat{\mathbf{A}}_{rij}$ is based on $\widehat{\mathbf{A}}_{ij}$. With the node embeddings $\mathbf{U}^{(0)}$ and the adjacency matrix $\widehat{\mathbf{A}}_{rij}$ as inputs, we apply RGCNs to extract high-dimensional representations of nodes. The computation for node u_i at the k -th layer in a relational multigraph under relation r can be derived as follows:

$$\mathbf{u}_i^{(k)} = \sigma\left(\sum_{r \in R_t} \sum_{j \in U_r} \widehat{\mathbf{A}}_{rij} \mathbf{W}_r^{(k)} \mathbf{u}_j^{(k-1)} + \mathbf{W}_0^{(k)} \mathbf{u}_i^{(k-1)}\right) \quad (2)$$

where \mathbf{U}_r is the set of neighbors of node u_i connected by relation of type r , $\mathbf{W}_0^{(k)}$ represents the learnable weight matrix for the root transformation in layer k , $\mathbf{W}_r^{(k)}$ is respectively the weight matrix of the k -th layer and $\sigma(\cdot)$ is an activation function (e.g., *RELU*). According to the category of each node, we divide the output representations $\mathbf{u}^{(k)} \in \mathbb{R}^{m \times d_u}$ into temporal relation labels representations $\mathbf{R} \in \mathbb{R}^{m_r \times d_u}$ and event attributes representations $\mathbf{T} \in \mathbb{R}^{m_t \times d_u}$.

2.4 Syntactic Graph Transformer

Temporal information related to events in sentences primarily originates from their SDP and surrounding context. To learn more relevant temporal information, we first acquire the SDP with the neighbor nodes of event pairs from the input layer. Then, we utilize a syntactic graph transformer encoder to obtain comprehensive representations of nodes from the SDP.

Input Layer Given an event pair $\langle e_i, e_j \rangle$ from one or two sentences, we apply a public dependency parser¹ to parse each sentence into syntactic tree graph that has only one root node. To connect the tree graphs of two sentences, we follow previous work [14] to establish a link between the root nodes of two different sentences and assign "cross-sentence" to their dependency relationship. For each node in the syntactic graph \mathcal{G}_s , we apply the RoBERTa [26] encoder to get word contextual represented vectors for initializing node representations. We use $\mathcal{P}_{ij} = \mathcal{P}_{iu} + \mathcal{P}_{uj}$ as the SDP from \mathbf{n}_i to \mathbf{n}_j , where u is the syntactic graph root node. Then, we add neighbor nodes of event pairs from the syntactic graph to \mathcal{P}_{ij} .

Encoding Layer Following graph transformer model [20, 32, 33], the graph encoder consists of a stack of identical graph layers. To learn rich node representations in \mathcal{P}_{ij} , we adopt the multi-head graph attention. In our approach, the embedding of each node \mathbf{n}_i by RoBERTa encoder as the initial hidden state of node h_i^0 and the node representation in our graph transformer is \mathbf{h}_i^{l-1} . Specifically, it allows each node to have its out edges and in edges with edge attribute denoting syntactic dependency to deal with the triples $(n_i, r_{d_{ij}}, n_j)$ as follow:

$$\mathbf{R}_{ij}^{l-1} = \left(\mathbf{h}_i^{l-1} \parallel \mathbf{r}_{d_{ij}}^{l-1} \parallel \mathbf{h}_j^{l-1} \right) \mathbf{W}_d + \mathbf{b}_d \quad (3)$$

¹ <https://spacy.io/api/dependencyparser>

where $r_{d_{ij}}^{l-1} \in \mathcal{R}_d$ is the edge label, $(\mathbf{h}_i^{l-1} \parallel \mathbf{r}_{d_{ij}}^{l-1} \parallel \mathbf{h}_j^{l-1})$ is the concatenation of the triples, $\mathbf{W}_d \in \mathbb{R}^{3d_{model}}$ and $b_d \in \mathbb{R}^{d_{model}}$. The multi-head attention mechanism builds on scaled dot-product attention, which operates on a package of queries \mathbf{Q}_i , keys \mathbf{K}_i , and values \mathbf{V}_i based on each node n_i as follow:

$$\text{Attention}(\mathbf{Q}_i, \mathbf{K}_i, \mathbf{V}_i) = \text{softmax}\left(\frac{\mathbf{Q}_i \mathbf{K}_i^T}{\sqrt{d_k}}\right) \mathbf{V}_i \quad (4)$$

where $\sqrt{d_k}$ is the scaling factor denoting the dimension size of each key vector. To perform self-attention, we begin with a linear transformation to generate a query vector \mathbf{Q}_i based on each node n_i and employ another two linear transformations to obtain the key vectors \mathbf{K}_i and value vectors \mathbf{V}_i based on another node in the triplets associated with node n_i . Subsequently, we conduct self-attention across all relevant triples, and compute a new node representation with multiple attention heads as shown below:

$$\mathbf{z}_i^l = \prod_x^{d_h} (\text{head}_i^x) \mathbf{W}_o \quad (5)$$

$$\text{head}_i^x = \text{softmax}\left(\frac{\mathbf{h}_i^{l-1} \mathbf{W}_q^x (\mathbf{r}_{ij}^{l-1} \mathbf{W}_k^x)^T}{\sqrt{d_k}}\right) \mathbf{R}_{ij}^{l-1} \mathbf{W}_v^x \quad (6)$$

where \mathbf{z}_i^l is the output of the multi-head graph attention for node n_i with d_h heads at l -th layer, \prod denotes the concatenation of the d_h attention heads, and $\mathbf{W}_q^x, \mathbf{W}_k^x, \mathbf{W}_v^x, \mathbf{W}_o$ is a learnable parameter, respectively. The fully connected feedforward network layer is applied individually and uniformly to each node. For each layer, we utilize residual connection followed by LayerNorm to obtain the final representations for each node.

2.5 Constraint Fusion Module

Given an event pair $\langle e_i, e_j \rangle$, a syntactic graph \mathcal{G}_s , and a raw constraint graph \mathcal{G}_c , we obtain events token representations \mathcal{Q} and temporal relation representations \mathbf{R} and event attribute representations \mathbf{T} by two aforementioned modules. To aggregate the outputs of the above two modules, we first construct new event representations and constraint representations and then apply the fusion operation over these to compute the final prediction.

Representation construction For each constraint $(t_{e_i}, r_{t_{ij}}, t_{e_j}) \in \mathcal{C}$ indicates a constraint rule. The new event representation is achieved by concatenating the representations of syntactic event and its corresponding event attribute representations as $\mathbf{o}_{ij} = (\mathbf{q}_{ij} \parallel \mathbf{t}_{e_i} \parallel \mathbf{t}_{e_j})$, where $\mathbf{q}_{ij} \in \mathcal{Q}$, $\mathbf{t}_{e_i}, \mathbf{t}_{e_j} \in \mathbf{T}$. Then, we concatenate the event attribute representations and their temporal relation representations as $\mathbf{c}_{t_{ij}} = (\mathbf{r}_{t_{ij}} \parallel \mathbf{t}_{e_i} \parallel \mathbf{t}_{e_j})$ to obtain the constraint representations, where $\mathbf{r}_{t_{ij}} \in \mathbf{R}$.

Fusion Layer Our fusion layer combines new event representations and constraint representations to calculate the final output. Intuitively, the similarity between the former parts (i.e., \mathbf{q}_{ij} and $\mathbf{r}_{t_{ij}}$) measures the semantic matching between the event pairs and the temporal relation. Then the final representation can be computed as follows:

$$\mathbf{g}_{ij} = \mathbf{o}_{ij} \mathbf{c}_{t_{ij}} \quad (7)$$

Finally, we feed the final representation \mathbf{g}_{ij} into a softmax classifier to calculate the probability distribution over relation labels as follows:

$$\hat{\mathbf{y}}_{ij} = \text{softmax}(\mathbf{W}_t \mathbf{g}_{ij} + \mathbf{b}_t) \quad (8)$$

where $\hat{\mathbf{y}}_{ij}$ denotes the probabilities over all possible temporal relations between event mentions e_i and e_j , \mathbf{W}_t is the weight of the classifier and \mathbf{b}_t is the bias. The cross-entropy loss function to train our model is:

$$\mathcal{L} = -\sum_{ij} \log \hat{\mathbf{y}}_{ij} (r_{t_{ij}} | \mathcal{Q}; \mathcal{G}_s; \mathcal{G}_c; \delta) \quad (9)$$

where δ is the set of parameters. During the test stage, the ground-truth temporal relation labels $r_{t_{ij}}$ is unknown. Therefore, we apply all constraint representations to calculate the posterior probabilities of the temporal relation and select the temporal relationship with the highest probability as the predicted result.

3 Experiment

3.1 Experimental Setup

Datasets We conduct our experiments on two public well-known benchmarks for ETRE, MATRES [28] and TB-DENSE [29]. MATRES contains refined annotations on TimeBank [30], Aquaint [34] and Platinum [35] documents. It annotates temporal relations only based on start time points along with four label types: *Before*, *After*, *Equal*, and *Vague*. On the other hand, TB-DENSE is a densely annotated dataset from TimeBank. In addition to the four label types used in MATRES, it introduces two additional label types: *Includes* and *Is_Included*. For both benchmark datasets, we use the same (train/dev/test) splits as previous studies [10, 15]. The detailed statistics of the two datasets can be found in Table 2.

Table 2. Data Statistics for MATRES and TB-DENSE

Dataset		Train	Dev	Test
MATRES	# of Docs	218	37	20
	# of Relations	11820	920	837
TB-DENSE	# of Docs	22	5	9
	# of Relations	5628	717	1785

Event Types TimeML [36] provides a definition of events as situations that happen or occur, or elements describing states or circumstances in which something obtains or holds the truth. The TimeML guidelines further classify events into seven distinct classes as follows. The definition and distribution of seven event types in MATRES and TB-DENSE are illustrated in Table 3.

Table 3. Event Type Definition and Distribution for MATRES and TB-DENSE

TYPES	DEFINITION	MATRES		TB-DENSE	
Occurrence	Describing something that happens(e.g., arrive)	3111	51.0%	849	56.8%
Reporting	Action of a person or organization declaring	1553	25.5%	191	12.8%
I_Action	Intensional action(e.g., attempt, try, promise)	549	9.0%	190	12.7%
I_State	Intensional state(e.g., believe, intend, want)	357	5.9%	107	7.2%
State	Circumstance where truth holds(e.g., war)	307	5.0%	80	5.3%
Aspectual	Aspectual prediction of event(e.g., stop, begin)	176	2.9%	61	4.1%
Perception	Physical perception of event(e.g., see, hear)	45	0.7%	16	1.1%

Evaluation Metrics For fair comparisons with previous research, we follow the same evaluation metrics as [20, 28], disregarding the *Vague* relation label from both datasets. We compute the precision, recall, and micro-average F1 score to maintain consistency with the baselines.

Hyper-parameters Settings In the experiment, we choose the pre-trained RoBERTa-large² to encode the input and optimize our model with BertAdam. The encoder layer size is 12 and the number of heads is 8 in the graph transformer. In RGCNs, we set graph embedding size as 220, hidden size as 250, output size as 300, layer size as 2, head number as 8, and edge type as 2. Through the hyper-parameter search in the experiment, we find the best performance can be achieved with the following settings: total training epochs is 5, a learning rate of 4e-6, and a batch size of 16. To prevent overfitting, we apply dropout with a rate of 0.5 before the classifier layer. Furthermore, all weight matrixes are initialized by Xavier initialization.

3.2 Results

Table 4 shows the performance of our approach CGSE and baseline methods for ETRE in each benchmark dataset, organized based on publication time. We employ the pre-trained RoBERTa-large for fine-tuning and optimizing our model with BertAdam. For these datasets, the following baselines are chosen for comparison.

² <https://huggingface.co/roberta-large>

Table 4. Model Performance on MATRES and TB-DENSE.

Models	MATRES			TB-DENSE		
	Pre	Recall	F1	Pre	Recall	F1
Structrued-Joint	59.0	60.2	59.6	52.6	46.5	49.4
SP+ILP	71.3	82.1	76.3	58.4	58.4	58.4
Deep-Structured	77.4	86.4	81.7	62.7	58.9	62.5
UAST	76.6	84.9	80.5	64.3	64.3	64.3
SGT	-	-	80.3	-	-	67.1
Relative-SenTime	<u>80.1</u>	84.4	82.2	66.5	66.5	66.5
PIPER	-	-	81.8	<u>67.2</u>	<u>68.3</u>	67.7
LEAF	-	-	82.1	-	-	66.7
UNIFIED	-	-	82.6	-	-	<u>68.1</u>
Bayesian-Trans	79.6	86.0	<u>82.7</u>	-	-	65.0
Our Method	81.7	85.0	83.3	68.4	68.4	68.4

- **Structrued-Joint** [10]: A joint event and temporal relation extraction model with shared representation learning to make predictions on events and relations simultaneously.
- **SP+ILP** [11]: A structured learning approach with Integer linear programming (ILP) methods to enforce global consistency.
- **Deep-Structured** [12]: A deep structured learning framework based on a structured support vector machine to encode the structure knowledge and learn long-range features.
- **UAST** [18]: An uncertainty-aware self-training framework to quantify the model uncertainty for tackling pseudo-labeling errors.
- **SGT** [20]: A graph transformer over dependency parse trees to represent the connection between an event pair in a document.
- **Relative-SenTime** [21]: A multi-task learning framework utilizing the relative sentence time to enhance cross-sentence temporal relation extraction.
- **PIPER** [22]: A logic-driven deep contrastive optimization pipeline for the event temporal relation extraction based on the designed hierarchical graph distillation network and the rule-match features.
- **LEAF** [23]: A simple and effective approach to extract rich temporal knowledge from unannotated corpora using diverse temporal knowledge patterns and pre-train a language model to inject this knowledge.
- **UNIFIED** [24]: A unified framework transforming temporal relations into logical expressions of time points and predicting the relations between certain time point pairs.
- **Bayesian-Trans** [25]: A bayesian learning-based method that models the temporal relation representations as latent variables and infers their values via bayesian inference and translational functions.

Overall, we observe that our method significantly outperforms baseline systems, although they are also based on the pre-trained model on MATRES and TB-Dense. Our CGSE has a 0.6 % F1 improvement on the MATRES dataset and a 0.3 % F1 improvement on the TB-Dense dataset, which indicates the effectiveness of our proposed CGSE model for the ETRE task.

3.3 Ablation Study

To better understand the effectiveness of each component of our model, we further conduct ablation studies to compare the performance of our approach with different configurations. In Table 5, CGSE -w/o constraint graph represents our method CGSE excluding the constraint graph. For CGSE -w syntactic graph, we replace the syntactic graph transformer with pre-trained RoBERTa.

Table 5. Performance of Different Models on MATRES

Model	Pre	Recall	F1
RoBERTa-F	79.6	81.2	80.4
CGSE -w/o constraint graph	80.2	82.3	81.3
CGSE -w syntactic graph	81.0	83.8	82.4
CGSE	81.7	85.0	83.3

The result demonstrates that each component of CGSE contributes significantly to our model's performance, as removing any of these components leads to a notable decrease in the F1 score. Removing the constraint graph in our model obviously has a profound impact on the F1 performance. Both the external information of event attributes and relation constraints provide significant improvements to our model. In addition, the syntactic graph is indispensable for reducing irrelevant information in complex contexts and gaining more useful temporal information. It is necessary to consider the internal syntactic structure of the sentence for a deeper temporal analysis. Most syntactic structures, such as parallel structures, infinitive structures, and adverbial clauses often exhibit a temporal order. The ablative experiment demonstrates that the syntactic graph transformer can effectively learn the syntactic structure features and context information related to events.

Impact of Constraint Graph We propose the constraint graph that enables our model to learn the inherent tense and type information of events and the temporal relationship label information. As depicted in Table 6, adding the constraint graph has improved the F1 values in each temporal label category, except for 'Equal'. This is attributed to the insufficient number of 'Equal' samples, as demonstrated in Table 7, resulting in inadequate learning. The RoBERTa-F baseline exhibits a significant discrepancy in F1 values between the 'Before' and 'After' temporal relations, which indicates inadequate learning for the "After" relation. Nevertheless, the constraint graph significantly improves this aspect.

Table 6. F1 Values of Different Variants on MATRES

Model	Before	After	Equal	Vague
RoBERTa-F	84.2	78.7	0.0	28.4
RoBERTa-F + Constraint	85.0	82.8	0.0	29.9
CGSE	85.9	83.7	0.0	31.5

Table 7. Temporal Label Distribution for MATRES and TB-DENSE

LABELS	MATRES		TB-DENSE	
	Before	6885	50.7%	2590
After	4575	33.7%	2104	16.5%
Equal	471	3.5%	215	1.7%
Includes	-	-	836	6.6%
Is_Included	-	-	1060	8.3%
Vague	1642	12.1%	5910	46.5%

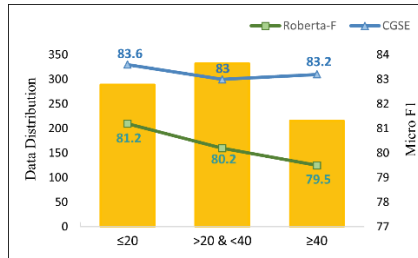
Based on our observations, at least three kinds of information in constraint graphs are beneficial. Firstly, event types and tenses offer external event information for the model to understand events. Secondly, constraint information provides direct and efficient prior temporal information for the model. Lastly, interactive information exists within constraint graphs, as temporal relation label nodes are indirectly linked through event attribute nodes. In such cases, we can employ message passing between nodes to transfer rich knowledge and capture intrinsic dependencies.

Impact of Syntactic Graph In this section, we conduct comparative experiments on the range of dependency path choices in syntactic graphs constructed from syntactic dependency trees. By selecting three kinds of dependency ranges, we compare the All Path (all syntactic dependency paths), the SDP Path (the shortest dependency path between two events), and the Neighbor Path (dependency paths directly connected to the event triggers) with our model (SDP Path + Neighbor Path). According to Table 8, the SDP Path is less effective than the Neighbor Path, as the Neighbor Path retains more context information relevant to events. The SDP Path mainly extracts syntactic structures between events, but not all syntactic structures are capable of expressing temporal relations. On the other hand, the All Path tends to include an excessive amount of irrelevant information, which undermines the performance of the model. Therefore, considering the SDP Path and the Neighbor Path at the same time can better capture both syntactic and contextual information.

Table 8. A Comparison of Different Ranges of Syntactic Dependency Path on MATRES

Model	Pre	Recall	F1
All Path	80.4	82.7	81.6
SDP Path	80.7	83.3	82.0
Neighbor Path	81.1	84.1	82.6
CGSE(Our Model)	81.7	85.0	83.3

Impact of Distance between Events We further analyze the impact of distance between events on the RoBERTa-F baseline and our approach. We categorize distances into three categories using values of 20 and 40, as shown in Fig. 3. The dataset distribution for these three categories is 289, 332, and 216 in the MATRES dataset, respectively. Based on our experimental results, we note that the RoBERTa-F model cannot perform well in predicting the temporal relation between two event mentions when they are far apart in sentences. Contrarily, our model demonstrates better with only minor variations across different distances. Our model declines more at distances of 20-40 due to a larger number of datasets, especially for cross-sentence events. This highlights the superiority and benefits of our approach in long-distance event pairs.

**Fig. 3.** Distance between event pairs on MATRES. The x-axis shows the number of subtokens between the two events mentions.

3.4 Error Analysis

To facilitate a comprehensive understanding of the output errors generated by our approach, we categorize them into three categories and provide a detailed analysis as Fig. 4 shows: (i) **Ambiguous context**. Due to some sentences not being clearly expressed, it is difficult for our model to comprehend the intricate and ambiguous context in which to infer temporal relations, as exemplified by S1. (ii) **Common sense knowledge**. Our approach cannot correctly distinguish between the actual events and the event containing common sense knowledge, as shown in S2. Furthermore, in most cases, the temporal relation among common sense events is annotated as *Vague*. (iii) **Imbalanced Labels**. We observe that accurately predicting the equal relation in both datasets is challenging. This difficulty arises from the low percentage of equal relation, as

presented in Table 7. In the MATRES dataset, the percentage of Before and After relations exceeds 80%. Meanwhile, in the TB-Dense dataset, the label distribution is even more imbalanced as the percentage of Vague relation is close to 50%, while the percentage of "Equal", "Includes" and "Includes_In" relations are 1.7% and less than 7% respectively.

S1:	The call, which happened as President wrapped up his first presidential visit to Israel, was an unexpected outcome from a Mideast trip that seemed to yield few concrete steps. <i>(Gold label: Before, Prediction label: After)</i>
S2:	Only Ireland, still struggling to recover from the banking collapse that required an international bailout in 2010, has a higher debt-to-G.D.P. As debts in Europe mount in inverse proportion to the ability of its citizens. <i>(Gold label: Vague, Prediction label: Before)</i>
S3:	Senator Patty Murray is the first chairwoman of the Budget Committee and is charged with shaping the Democratic strategy in the fiscal battle dominating Capitol Hill. <i>(Gold label: Equal, Prediction label: Before)</i>

Fig. 4. Error analysis. Event mentions are highlighted in blue and green. Each line following the sentence is its temporal relation label.

4 Related Work

Most early traditional studies learn the event attributes based on pattern matching [37, 38] and statistical machine learning [39-41]. Mani et.al [39] build a MaxEnt classifier relying on event attribute features that were hand-tagged in the corpus. Chambers [40] describes automatic machine learning using event attributes features. However, how to make use of these features has been under-explored since the emergence of neural methods. To fill this gap, our approach introduces a novel constraint graph to learn event attribute features in ETRE tasks to represent temporal dependencies better.

Recently, neural networks [5-9, 11-13] and large-scale pre-trained language models [10, 14-21] have been employed for ETRE, which have achieved impressive results. Several of these studies have also explored the shortest dependency paths between two events for ETRE. Meng [6] utilizes a streamlined LSTM-based architecture and Cheng [14] adopts BiLSTM along shortest dependency paths to capture temporal relation. Mathur et al. [19] show that graph-based neural networks can learn highly relational data relationships in graphs incorporating syntactic features, discourse features, and temporal arguments from semantic role labels. Our approach is similar to the recent work [20], which performs a graph transformer with dependency parse trees and adopts a much more complex attention machine. In contrast, our approach differs in that we utilize another pre-trained model that is more suitable for capturing temporal series information. Thus, we employ the graph transformer with a simplified structure to obtain the most meaningful temporal information related to events.

5 Conclusion

In this paper, we propose a novel approach, Constraint Graph-based and Syntax-aware Event Temporal Relation Extraction (CGSE). Our method obtains more event and temporal information through the constraint graph and syntactic graph. Furthermore, our method effectively addresses the complex contextual challenges arising from event pairs that are widely distributed in the text. Experiments on benchmark datasets show that our approach is significantly superior to previous state-of-the-art methods. In the future, we aim to discover an effective graph structure and investigate the potential of our approach in other relation extraction tasks.

References

1. Do, Q., Lu, W., Roth, D.: Joint inference for event timeline construction. In: Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning. pp. 677–687 (2012)
2. Chaturvedi, S., Peng, H., Roth, D.: Story comprehension for predicting what happens next. In: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing. pp. 1603–1614 (2017)
3. Khashabi, D., Khot, T., Sabharwal, A., Roth, D.: Question answering as global reasoning over semantic abstractions. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 32 (2018)
4. Yao, L., Peng, N., Weischedel, R., Knight, K., Zhao, D., Yan, R.: Plan-and-write: Towards better automatic storytelling. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 33, pp. 7378–7385 (2019)
5. Xu, Y., Mou, L., Li, G., Chen, Y., Peng, H., Jin, Z.: Classifying relations via long short term memory networks along shortest dependency paths. In: Proceedings of the 2015 conference on empirical methods in natural language processing. pp. 1785–1794 (2015)
6. Meng, Y., Rumshisky, A., Romanov, A.: Temporal information extraction for question answering using syntactic dependencies in an lstm-based architecture. In: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing. pp. 887–896 (2017)
7. Choubey, P.K., Huang, R.: A sequential model for classifying temporal relations between intra-sentence events. In: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing. pp. 1796–1802 (2017)
8. Ning, Q., Wu, H., Peng, H., Roth, D.: Improving temporal relation extraction with a globally acquired statistical resource. In: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers). pp. 841–851 (2018)
9. Ning, Q., Subramanian, S., Roth, D.: An improved neural baseline for temporal relation 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). pp. 6203–6209 (2019)
10. Han, R., Ning, Q., Peng, N.: Joint event and temporal relation extraction with shared representations and structured prediction. arXiv preprint arXiv:1909.05360 (2019)
11. Ning, Q., Feng, Z., Roth, D.: A structured learning approach to temporal relation extraction. arXiv preprint arXiv:1906.04943 (2019)

12. Han, R., Hsu, I., Yang, M., Galstyan, A., Weischedel, R., Peng, N., et al.: Deep structured neural network for event temporal relation extraction. *arXiv preprint arXiv:1909.10094* (2019)
13. Dai, Q., Kong, F., Dai, Q.: Event temporal relation classification based on graph convolutional networks. In: *CCF International Conference on Natural Language Processing and Chinese Computing*. pp. 393–403. Springer (2019)
14. Cheng, F., Miyao, Y.: Classifying temporal relations by bidirectional lstm over dependency paths. In: *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. pp. 1–6 (2017)
15. Han, R., Ren, X., Peng, N.: Econet: Effective continual pretraining of language models for event temporal reasoning. In: *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. pp. 5367–5380 (2021)
16. Zhou, Y., Yan, Y., Han, R., Caufield, J.H., Chang, K.W., Sun, Y., Ping, P., Wang, W.: Clinical temporal relation extraction with probabilistic soft logic regularization and global inference. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. vol. 35, pp. 14647–14655 (2021)
17. Wen, H., Ji, H.: Utilizing relative event time to enhance event-event temporal relation extraction. In: *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. pp. 10431–10437 (2021)
18. Cao, P., Zuo, X., Chen, Y., Liu, K., Zhao, J., Bi, W.: Uncertainty-aware self-training for semi-supervised event temporal relation extraction. *Proceedings of the 30th ACM International Conference on Information & Knowledge Management* (2021), <https://api.semanticscholar.org/CorpusID:240230593>
19. Mathur, P., Jain, R., Dernoncourt, F., Morariu, V., Tran, Q.H., Manocha, D.: Timers: document-level temporal relation extraction. In: *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*. pp. 524–533 (2021)
20. Zhang, S., Ning, Q., Huang, L.: Extracting temporal event relation with syntax-guided graph transformer. In: *Findings of the Association for Computational Linguistics: NAACL 2022*. pp. 379–390 (2022)
21. Xie, P., Zhu, X., Zhang, C., Hu, Z., Yang, G.: Cross-sentence temporal relation extraction with relative sentence time. In: *International Conference on Knowledge Science, Engineering and Management*. pp. 346–357. Springer (2022)
22. Zhang, B., Li, L.: Piper: A logic-driven deep contrastive optimization pipeline for event temporal reasoning. *Neural Networks* 164, 186–202 (2023)
23. Lim, S., Yin, D., Peng, N.: Leaf: Linguistically enhanced event temporal relation framework. In: *Proceedings of the 2nd Workshop on Pattern-based Approaches to NLP in the Age of Deep Learning*. pp. 6–19 (2023)
24. Huang, Q., Hu, Y., Zhu, S., Feng, Y., Liu, C., Zhao, D.: More than classification: A unified framework for event temporal relation extraction. *arXiv preprint arXiv:2305.17607* (2023)
25. Tan, X., Pergola, G., He, Y.: Event temporal relation extraction with bayesian translational model. *arXiv preprint arXiv:2302.04985* (2023)
26. Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., Stoyanov, V.: Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692* (2019)
27. Schlichtkrull, M., Kipf, T.N., Bloem, P., Van Den Berg, R., Titov, I., Welling, M.: Modeling relational data with graph convolutional networks. In: *The Semantic Web: 15th International Conference, ESWC 2018, Heraklion, Crete, Greece, June 3–7, 2018, Proceedings* 15. pp. 593–607. Springer (2018)

28. Ning, Q., Wu, H., Roth, D.: A multi-axis annotation scheme for event temporal relations. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). pp. 1318–1328 (2018)
29. Cassidy, T., McDowell, B., Chambers, N., Bethard, S.: An annotation framework for dense event ordering. In: Annual Meeting of the Association for Computational Linguistics (2014), <https://api.semanticscholar.org/CorpusID:7294125>
30. Llorens, H., Saquete, E., Navarro, B.: Timeml events recognition and classification: learning crf models with semantic roles. In: Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010). pp. 725–733 (2010)
31. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 (2016)
32. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. *Advances in neural information processing systems* 30 (2017)
33. Wang, T., Wan, X., Jin, H.: Amr-to-text generation with graph transformer. *Transactions of the Association for Computational Linguistics* 8, 19–33 (2020)
34. Graff, D.: The acquaint corpus of english news text:[content copyright] portions© 1998-2000 new york times, inc.,© 1998-2000 associated press, inc.,© 1996-2000 xinhua news service. *Linguistic Data Consortium* p. 9 (2002)
35. UzZaman, N., Llorens, H., Derczynski, L., Allen, J., Verhagen, M., Pustejovsky, J.: Semeval2013 task 1: Tempeval-3: Evaluating time expressions, events, and temporal relations. In: Second Joint Conference on Lexical and Computational Semantics (* SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013). pp. 1–9 (2013)
36. Pustejovsky, J., Castano, J.M., Ingria, R., Sauri, R., Gaizauskas, R.J., Setzer, A., Katz, G., Radev, D.R.: Timeml: Robust specification of event and temporal expressions in text. *New directions in question answering* 3, 28–34 (2003)
37. Passonneau, R.J.: A computational model of the semantics of tense and aspect. *Computational Linguistics* 14(2), 44–60 (1988)
38. Hitzeman, J., Moens, M., Grover, C.: Algorithms for analysing the temporal structure of discourse. In: Seventh Conference of the European Chapter of the Association for Computational Linguistics (1995)
39. Mani, I., Schiffman, B., Zhang, J.: Inferring temporal ordering of events in news. In: Companion Volume of the Proceedings of HLT-NAACL 2003-Short Papers. pp. 55–57 (2003)
40. Chambers, N., Wang, S., Jurafsky, D.: Classifying temporal relations between events. In: Proceedings of the 45th annual meeting of the association for computational linguistics companion volume proceedings of the demo and poster sessions. pp. 173–176 (2007)
41. Mirza, P., Tonelli, S.: Classifying temporal relations with simple features. In: Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics. pp. 308–317 (2014)

A Detailed Constraints

In this section, we present detailed constraints on the event types utilized in constructing the constraint graph. The temporal co-occurrence frequency of event type pairs is illustrated in Table 9.

Table 9. Frequency of Event Type Pairs Co-occurrence on Temporal Relations

Event Type Pair		Before	After	Equal	Vague
Occurrence	Reporting	1395	99	38	92
Reporting	Occurrence	277	1195	185	47
Occurrence,	I_Action	312	156	17	68
I_Action	Occurrence	295	209	21	56
Occurrence	I_State	206	85	10	52
I_State	Occurrence	119	191	11	47
Occurrence	State	115	133	15	36
State	Occurrence	168	100	19	50
Occurrence	Aspectual	76	49	6	21
Aspectual	Occurrence	106	61	9	29
Occurrence	Perception	31	24	2	6
Perception	Occurrence	25	34	0	9
Reporting	I_Action	71	183	14	38
I_Action	Reporting	235	22	7	24
Reporting	I_State	60	159	16	33
I_State	Reporting	155	10	2	19
Reporting	State	19	167	2	29
State	Reporting	148	5	2	9
Reporting	Aspectual	21	54	0	16
Aspectual	Reporting	81	9	4	4
Reporting	Perception	5	24	1	5
Perception	Reporting	28	1	1	1
I_Action	I_State	28	25	3	9
I_State	I_Action	30	19	3	11
I_Action	State	23	32	0	10
State	I_Action	47	10	1	3
I_Action	Aspectual	22	11	3	4
Aspectual	I_Action	21	6	0	7
I_Action	Perception	3	3	0	0
Perception	I_Action	5	1	0	1
I_State	State	8	33	3	1
State	I_State	24	5	4	5
I_State	Aspectual	8	7	0	3
Aspectual	I_State	11	2	0	2
I_State	Perception	6	0	2	2
Perception	I_State	2	1	0	1
State	Aspectual	8	2	1	2
Aspectual	State	7	10	1	3
State	Perception	1	4	2	1
Perception	State	3	4	0	1
Aspectual	Perception	3	0	0	0
Perception	Aspectual	1	1	0	0