Discourse-Level Event Temporal Ordering with Uncertainty-Guided Graph Completion

Jian Liu, **Jinan Xu**, **Yufeng Chen** and **Yujie Zhang**
Beijing Jiaotong University, School of Computer and Information Technology, China
jianliu@bjtu.edu.cn, jaxu@bjtu.edu.cn, chenyf@bjtu.edu.cn, yjzhang@bjtu.edu.cn

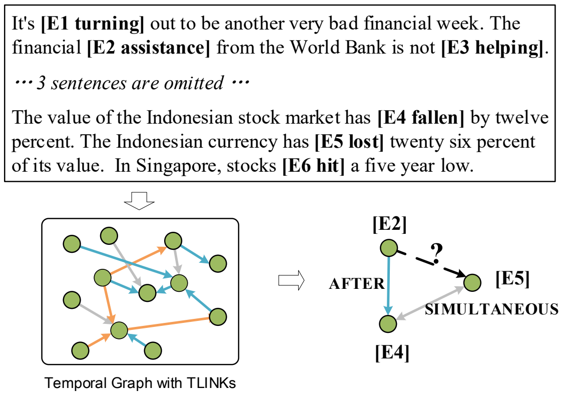
Abstract

Learning to order events at discourse-level is a crucial text understanding task. Despite many efforts for this task, the current state-of-the-art methods rely heavily on manually designed features, which are costly to produce and are often specific to tasks/domains/datasets. In this paper, we propose a new graph perspective on the task, which does not require complex feature engineering but can assimilate global features and learn inter-dependencies effectively. Specifically, in our approach, each document is considered as a temporal graph, in which the nodes and edges represent events and event-event relations respectively. In this sense, the temporal ordering task corresponds to constructing edges for an empty graph. To train our model, we design a graph mask pre-training mechanism, which can learn inter-dependencies of temporal relations by learning to recover a masked edge following graph topology. In the testing stage, we design an certain-first strategy based on model uncertainty, which can decide the prediction orders and reduce the risk of error propagation. The experimental results demonstrate that our approach outperforms previous methods consistently and can meanwhile maintain good global consistency.

1 Introduction

Learning to order events at discourse-level is a crucial text understanding task, which is necessary for many applications including event timeline construction [Do *et al.*, 2012; Reimers *et al.*, 2016], time-aware summarization [Yan *et al.*, 2011], temporal commonsense reasoning [Zhou *et al.*, 2019], and others. Consider the example in Figure 1. Given a document marked with events, a system should assign a temporal link, i.e., TLINK [Allen, 1984], between every pair of events. For example, a TLINK of BEFORE should be assigned between [E2 assistance] and [E4 fallen], denoted by [E2] $\xrightarrow{\text{BEFORE}}$ [E4], indicating that [E2] occurs *before* [E4].

The challenge of discourse-level event temporal ordering derives from the appearance of long contexts. To succeed in the task, a system should be able to reason over global features and meanwhile maintain document-wide consistency.


It's **[E1 turning]** out to be another very bad financial week. The financial **[E2 assistance]** from the World Bank is not **[E3 helping]**.

··· 3 sentences are omitted ···

The value of the Indonesian stock market has **[E4 fallen]** by twelve percent. The Indonesian currency has **[E5 lost]** twenty six percent of its value. In Singapore, stocks **[E6 hit]** a five year low.

Figure 1: Illustration of framing discourse-level event temporal ordering as a graph completion problem, where the nodes and edges indicate events and their temporal relations respectively.

For example, if a model has predicted [E2] $\xrightarrow{\text{BEFORE}}$ [E4] and [E4] $\xleftrightarrow{\text{SIMULTANEOUS}}$ [E5], it should also figure out that [E2] $\xrightarrow{\text{BEFORE}}$ [E5]. It is generally difficult to learn such patterns from texts solely, and the state-of-the-art methods usually resort to human-designed rules [Do *et al.*, 2012; Ning *et al.*, 2017; Ning *et al.*, 2018; Han *et al.*, 2019a; Han *et al.*, 2019b]. Despite many progresses, these rules are usually costly to produce and often specific to tasks/domains/datasets, which limit the applicability of previous methods.

In this paper, we provide a new perspective for the event temporal ordering task — framing it as a *graph completion* problem. As noted in Figure 1 (Below), in our approach each document is considered as a temporal graph, in which each node has a one-to-one mapping relationship to an event. In this graph view, the event temporal ordering task corresponds to predicting/constructing the edges of the graph — which designate event-event temporal relations. By adopting this graph view, we can directly leverage the recent advances in graph representation learning [Chen *et al.*, 2020] to enhance training, which endows our model the ability of global reasoning and meanwhile mining the underlying interdependencies of temporal relations effectively.

We devise an uncertainty-guided graph completion (UCGraph) framework. For training, our framework employs a *graph mask pre-training* strategy, in which we randomly mask a portion of edges out and then learn to reconstruct them following the remaining graph topology. This graph view enables our model to assimilate document-level features for reasoning and meanwhile learn the sense of global logical consistency. Nevertheless, a gap may exist in the inference stage as we would begin with an empty graph (instead of a masked graph), and the early predictions can have great impact on the late predictions. To mitigate this problem, we design a "certain-first" strategy in UCGraph based on model uncertainty [Gal and Ghahramani, 2016], which can find the optimal edge prediction orders and therefore minimize error propagation in the graph completion process (§ 6.2).

To verify the effectiveness of our approach, we have conducted extensive experiments on the standard benchmark datasets [Naik *et al.*, 2019]. The experimental results demonstrate that our approach consistently outperforms previous methods and sets up a new state-of-the-art. In addition to its superior performance, our approach also show advantages over previous methods in maintaining global consistency, which is a crucial aspect of event temporal ordering. We have released our code at https://github.com/jianliu-ml/EventTemp to facilitate further exploration.

To summarize, our contributions are three-fold:

- This paper provides a new graph view on the task of discourse-level event temporal ordering, which can effectively assimilate document-level evidence for reasoning and learn inter-dependencies of temporal relations without complex feature engineering.
- We proposes a model consisting of graph mask pre-training and uncertainty-guided graph completion, which can learn inter-dependencies of temporal relations and find optimal prediction orders respectively. To our best knowledge, this is the first work introducing graph representation learning and uncertainty modeling to the task of discourse-level event temporal ordering.
- We set up a new state-of-the-art on the standard benchmark datasets, and we will release our code to facilitate following studies.

2 Related Work

Event Temporal Ordering. Temporal ordering of events is a crucial text understanding task. Shaped by the proposed TimeML annotation [Pustejovsky *et al.*, 2003a] and the related corpora such as TimeBank [Pustejovsky *et al.*, 2003b] and TimeBank-Dense [Cassidy *et al.*, 2014], most previous works focus on addressing *local* temporal relations, i.e., they focus on events in the same or adjacent sentences [Bethard *et al.*, 2007; Verhagen *et al.*, 2007; UzZaman and Allen, 2010; Chang and Manning, 2012; Chambers, 2013; Chambers *et al.*, 2014; Reimers *et al.*, 2016], Despite many progresses, the reliance on local features often restricts their ability to address *global* event relations. As a remedy, a few of works have explored introducing document-structure constraints [Ning *et al.*, 2017; Han *et al.*, 2019a], entity co-reference patterns [Do *et al.*, 2012], and event causal clues [Do *et al.*, 2012; Ning *et al.*, 2018] to learn global dependencies. Nevertheless, designing such rules requires substantial domain expertise, which may limit the applicability of previous works. Recently, focusing on global event temporal relations specifically, [Naik *et al.*, 2019] benchmark discourse-level event temporal ordering by proposing a new corpora TDDiscourse. As suggested by [Naik *et al.*, 2019], global event temporal ordering is very challenging, and the previous state-of-the-art systems underperforms a simple majority-class baseline.

Graph Representation Learning. Recent years have witnessed a surge in exploring graph representation learning [Chen *et al.*, 2020]. Graph neural networks, including Graph Convolution Network [Kipf and Welling, 2017] and its variants [Veličković *et al.*, 2018; Schlichtkrull *et al.*, 2018], have demonstrated state-of-the-art performance in addressing many tasks, including semantic role labeling [Marcheggiani and Titov, 2017], relation extraction [Zhang *et al.*, 2018], event extraction [Liu *et al.*, 2019], and others. Among all GCNs variants, Relational Graph Convolutional Networks (R-GCNs) [Schlichtkrull *et al.*, 2018] is a particular structure that facilitate relational reasoning. To our best knowledge, this is the first work introducing R-GCNs to the task of discourse-level event temporal ordering.

3 Approach

Figure 2 schematically visualizes the proposed UCGraph framework. Specifically, at the training time, a graph mask pre-training mechanism is employed to learn the underlying inter-dependencies of edges, by reconstructing an edge conditioned on the remaining graph. At the test time, UCGraph employs an uncertainty-guided graph completion strategy to rank each predicted edge and select the most uncertain one as current prediction. Let a document be D, and the event set be E_D. Let $\mathcal{R}$ be the TLINK set. We denote by $e_i \in E_D$ the ith event, and $r_{i,j} \in \mathcal{R}$ the TLINK between e_i and e_j. Our model UCGraph aims to structure D as a complete graph G_D, where the ith node corresponds to e_i, and $r_{i,j}$ is the edge connecting e_i and e_j. Considering the reflexivity of TLINKs (e.g., $e_i \xrightarrow{\text{BEFORE}} e_j$ always implies $e_i \xleftarrow{\text{AFTER}} e_j$), we assert $i < j$ [1]. In this way, the total number of edges in G_D is $\frac{|E_D| \times (|E_D|-1)}{2}$.

3.1 Graph Mask Pre-Training

To learn the underlying inter-dependencies of temporal relations, UCGraph employs a graph mask pre-training strategy.

Random Edge Masking. Given a training document D and its temporal graph, we first adopt a random mask strategy to exclude some edges[2] in G_D (inspired by the masked language modeling objective in BERT [Devlin *et al.*, 2019]), and then employ a graph model to re-construct these edges conditioned on the remaining graph. Assume a masked edges is $r_{i',j'}$, which connects two end nodes e'_i and e'_j. Then in the edge reconstruction phase, our goal is to recover $r_{i',j'}$ by exploring the remaining graph's structure. This graph formulation enables our model to learn the dependency of the masked edge on other edges, and assimilate global features for reasoning.

[1]Note this setting is in line with previous evaluations [Ning *et al.*, 2018; Naik *et al.*, 2019].

[2]The optimal portion is set to 5%, which is based on a grid search on the development set (c.f., § 6.1).

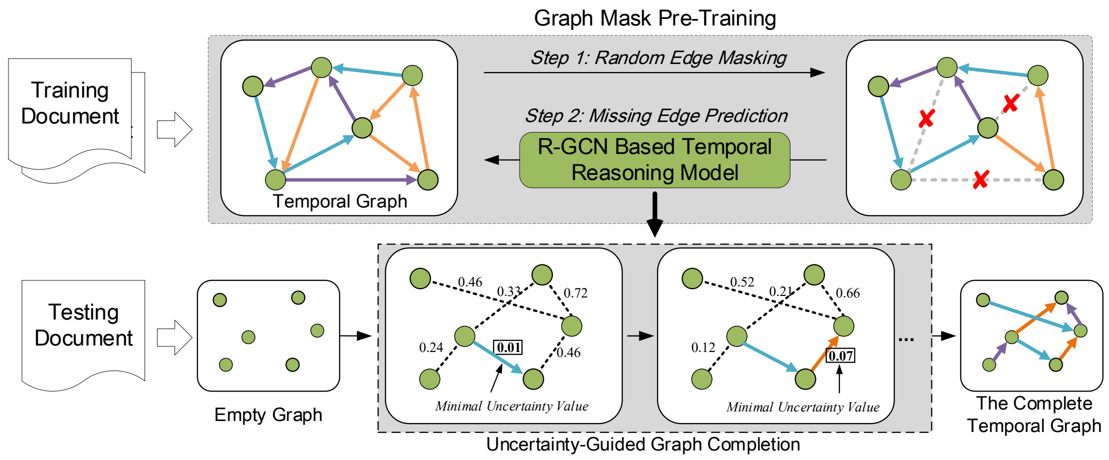

Graph Mask Pre-Training
Training Document
Step 1: Random Edge Masking
Step 2: Missing Edge Prediction
R-GCN Based Temporal Reasoning Model
Temporal Graph
Testing Document
Empty Graph
Minimal Uncertainty Value
Uncertainty-Guided Graph Completion
The Complete Temporal Graph

Figure 2: The overview of UCGraph. Up: A graph mask pre-training mechanism is adopted to learn the underlying inter-dependencies of edges. Down: An uncertainty-guided strategy is devised at the testing time to learn prediction orders for graph completion.

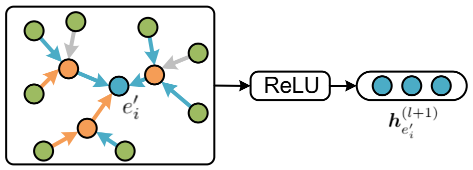


Figure 3: Graph representation learning via R-GCNs.

Graph Representation Learning via R-GCNs. To recover a masked edge $r_{i',j'}$ (with two end nodes e'_i and e'_j), we adopt Relational Graph Convolutional Networks (R-GCNs) [Schlichtkrull *et al.*, 2018], involving two major procedures:

1) **Node Representations Learning**, in which we first learn the *node representation* of each node in the graph. Particularly, we use the event's within-sentence representation as the initialized node representation, which is computed using Bi-directional LSTM (BiLSTM) [Hochreiter and Schmidhuber, 1997] and BERT [Devlin *et al.*, 2019]. Accordingly, the initialized node representation of e'_i is denoted by $\boldsymbol{h}^{(0)}_{e'_i}$.

2) **Graph Representations Learning**, in which we further learn the graph-aware representation of a node via R-GCNs. As shown in Figure 3, the graph-aware representation of e'_i (and e'_j) is learned by assimilating representations of its direct neighborhoods, and many R-GCNs layers are stacked to model long-range inter-dependencies. Particularly, the graph representation of e'_i at the l+1th layer is computed as:

$$\boldsymbol{h}^{(l+1)}_{e'_i} = \sigma(\sum_{r\in\mathcal{R}}\sum_{e_k\in\mathcal{N}^r(e'_i)}\frac{1}{c_{e'_i,r}}\boldsymbol{W}^{(l)}_r\boldsymbol{h}^{(l)}_{e_k}) \quad (1)$$

where σ denotes the ReLU activate function; r denotes a particular TLINK label[3] in the pre-defined label set $\mathcal{R}$; $\mathcal{N}^r(e'_i)$ denotes the set of neighbor nodes of e'_i under relation r; $c_{e'_i,r}$ is a normalization constant, and $\boldsymbol{W}^{(l)}_r$ is the parameter regarding r at the lth layer. The graph representation of e'_i is set as R-GCNs' output, denoted as $\boldsymbol{H}_{e'_i}$, and the representation of e'_j is computed in a similar way, denoted by $\boldsymbol{H}_{e'_j}$.

Edge Prediction and Optimization. Based on $\boldsymbol{H}_{e'_i}$ and $\boldsymbol{H}_{e'_j}$, we conduct a multi-class classification to predict the masked edge:

$$\boldsymbol{o}_{e'_i,e'_j} = \text{softmax}(\boldsymbol{W}_{out}[\boldsymbol{H}_{e'_i};\boldsymbol{H}_{e'_j}] + b_{out}) \quad (2)$$

where $\boldsymbol{o}_{e'_i,e'_j}$ is an output vector containing the predictive probabilities of different TLINK labels, and the predicted result is the label having the largest value in $\boldsymbol{o}_{e'_i,e'_j}$; [;] indicates concatenation computation; $\boldsymbol{W}_{out}$ and b_{out} are model parameters. We train our model with cross-entropy loss:

$$\mathcal{L}(\Theta) = -\sum_{(e'_i,e'_j)\in T}\mathrm{p}(\mathrm{r}_{e'_i,e'_j}|(\mathrm{e}'_i,\mathrm{e}'_j)) \quad (3)$$

where (e'_i, e'_j) ranges over each masked edge; $r_{e'_i,e'_j}$ denotes the ground-truth TLINK label; $\mathrm{p}(\mathrm{r}_{e'_i,e'_j}|(\mathrm{e}'_i,\mathrm{e}'_j))$ is the predictive probability of $r_{e'_i,e'_j}$ in $\boldsymbol{o}_{e'_i,e'_j}$. We adopt Adam rules [Kingma and Ba, 2015] for model optimization.

3.2 Uncertainty-Guided Graph Completion

A gap max exist between training and testing, as in the testing stage we should build a temporal graph from the ground up (i.e., all the edges are masked), and previous predictions affect the following predictions. To minimize error propagation, we propose a "certain-first" strategy to explore the edge prediction orders, as visualized in Figure 2.

Uncertainty Modeling. Assume our model's prediction is r for a masked edge. We next estimate our model's uncertainty of r via MC-Dropout [Gal and Ghahramani, 2016]. Specifically, we conduct K forward passes with dropout layers being activated and get K output values $\tilde{\boldsymbol{r}}$ = [$\tilde{r}_1$, $\tilde{r}_2$, ..., $\tilde{r}_K$][4]. According to [Gal and Ghahramani, 2016], the uncertainty of r empirically equals to the variance of $\tilde{\boldsymbol{r}}$. Considering r is a categorical value, we adopt Shannon's entropy to

[3]Including a NA label to denote "no-relation".

[4]The values may be different due to the activated dropout layers.

Algorithm 1 Certain-First Graph Completion

Input: A test document D annotated with a set of events E_D
Output: A complete graph G

```
1: Transfer $D$ as an empty graph $G$ with nodes being $E_D$
2: while $G$ is not completed do
3:     Predict TLINKs for all the missing edges in $G$
4:     Estimate uncertainties of TLINKs via Eq. (4)
5:     Select TLINK with minimal uncertainty value as current prediction, and insert the edge into $G$
6: end while
```

model the variance of $\tilde{\boldsymbol{r}}$:

$$\mathrm{Unc(r)} = -\sum_{\mathrm{i=1}}^{\mathrm{K}} \mathrm{p}(\tilde{\mathrm{r}}_{\mathrm{i}}) \log \mathrm{p}(\tilde{\mathrm{r}}_{\mathrm{i}}) \tag{4}$$

where $\mathrm{p}(\tilde{\mathrm{r}}_{\mathrm{i}}) = \frac{\mathrm{N}(\tilde{r}_i)}{\mathrm{K}}$ and $\mathrm{N}(\tilde{r}_i)$ is the frequency of $\tilde{r}_i$ in $\tilde{\boldsymbol{r}}$. Note lower $\mathrm{Unc(r)}$ indicates smaller uncertainty of r, which also implies a more reliable prediction.

Certain-First Graph Completion Strategy. We design a certain-first graph completion strategy to decide prediction orders, as shown in Algorithm 1. Specifically, our algorithm starts with an empty graph G, which corresponds to a test document, and then conducts the following steps:

- **Step 1**: Pseudo-predict TLINK labels for all the missing edges in G.
- **Step 2**: Estimate the uncertainty value for each edge prediction using E.q. (4).
- **Step 3**: Select an prediction with the minimal uncertainty value, and add the corresponding edge into G.
- **Step 4**: Repeat the above steps until G is completed.

Note in **Step 3**, when a new edge is added into the G, the graph structure will change, and thus the next predictions may be different from the current ones. The above algorithm will repeat $\frac{|E_D|\times(|E_D|-1)}{2}$ times exactly, which equals to the total number of edges in the graph[5]. We compare our method with different graph completion methods in § 6.2.

4 Experimental Setups

Datasets and Evaluation Metrics. We use TDDiscourse, the largest discourse-level event temporal ordering benchmark, as the test bed [Naik *et al.*, 2019]. It includes two subsets: 1) TDD-Man, which augments TimeBank-Dense (TBDense) [Cassidy *et al.*, 2014] by manually annotating TLINKs between event pairs that are more than one sentence apart. 2) TDD-Auto, which derives new TLINKs in the document with automatic inference rules. Table 1 and Table 2 compare the sizes and label distributions of TBDense, TDD-Man and TDD-Auto. We adopt Precision (P), Recall (R), and F1 score (F1) as estimation metrics, same as previous works to ensure comparability [Ning *et al.*, 2017; Naik *et al.*, 2019; Han *et al.*, 2019b].

[5]The computation complexity of our method (for inference) is $\mathrm{O(K \times N^2)}$, where K is the number of forward passes in uncertainty computing, and N is the number of events in a document.

Dataset	Train	Dev	Test
TBDense [Cassidy *et al.*, 2014]	4,032	629	1,427
TDD-Man [Naik *et al.*, 2019]	4,000	650	1,500
TDD-Auto [Naik *et al.*, 2019]	32,609	1,435	4,258

Table 1: Number of temporal relations in TBDense, TDD-Man, and TDD-Auto. In TBDense, only event-event TLINKs are counted.

Dataset	a	b	s	i	ii
TBDense	18%	22%	2%	5%	6%
TDD-Man	13%	27%	3%	38%	19%
TDD-Auto	28%	32%	16%	11%	13%

Table 2: Distribution of TLINKs in different datasets. Assume two events are e_1 and e_2. The TLINK of **a** defines e_1 occurs *after* e_2; **b** defines e_1 occurs *before* e_2; **s** defines e_1 occurs *simultaneously* as e_2; **i** defines e_1 *includes* e_2; **ii** defines e_1 *is included by* e_2.

Implementation Details. The hyper-parameters of our model are tuned on the development set of TDDiscourse. Finally, for graph mask pre-training, the mask portion is set as 5% (chosen from 1% to 50%, c.f., § 6.1). The number of R-GCN layers is set at 3 (chosen from [1, 2, 3, 4, 5]), and we use DEEP GRAPH LIBRARY[6] (DGL) to implement graph convolution algorithm. To learn the node representations, for BERT encoder, we use BERT-Base architecture; for BiLSTM encoder, we use Glove embeddings [Pennington *et al.*, 2014] and set the hidden dimension to 256 (chosen from [64, 128, 256, 512]). In uncertainty modeling, we set K to 20 to balance speed and efficiency. For testing, we consider event pairs which are 15 or fewer sentences apart following [Naik *et al.*, 2019] (The evaluation is not changed).

Baselines. We compare our method with the following baselines: **Majority**, which assigns the majority-class label to each event pair. Despite its simplicity, Majority outperforms most existing state-of-the-art methods on TDD-Man [Naik *et al.*, 2019]. **CAEVO** [Chambers *et al.*, 2014], a previous state-of-the-art method for identifying sentence-level TLINK which heuristic rules. **BiLSTM** [Cheng and Miyao, 2017], which introduces BiLSTM to learn representation of events to predict TLINKs. **SP+ILP** [Ning *et al.*, 2017], which adds global constraints via integer linear programming (ILP), aiming to mitigate the problem that CAEVO and BiLSTM make separate local decisions that may result in global inconsistency. **Deep SSVM** [Han *et al.*, 2019a], which leverages structured support vector machine to make global predictions. **Mult-Task** [Han *et al.*, 2019b], which jointly predict events and relations. **BERT**, which adopts BERT [Devlin *et al.*, 2019] to learn event representations (same as our method in learning node representations), but it conducts a pair-wise classification and ignores inter-dependencies. Our approach is denoted by UCGraph. We use UCGraph+BiLSTM and UCGraph+BERT to designate learning node representations via BiLSTM and BERT respectively.

[6]https://www.dgl.ai/

	TBDense			TDD-Auto			TDD-Man		
Method	**P**	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**
Major Voting	40.5	40.5	40.5	34.2	32.3	33.2	37.8	36.3	37.1
CAEVO [Chambers *et al.*, 2014]	49.9	46.6	48.2	61.1	32.6	42.5	32.3	10.7	16.1
SP [Ning *et al.*, 2017]	37.7	37.8	37.7	43.2	43.2	43.2	22.7	22.7	22.7
SP + ILP [Ning *et al.*, 2017]	58.4	58.4	58.4	46.4	45.9	46.1	23.9	23.8	23.8
Deep SSVM [Han *et al.*, 2019a]	-	-	63.2	$65.6^{\dagger}$	$53.3^{\dagger}$	$58.8^{\dagger}$	$41.2^{\dagger}$	$40.8^{\dagger}$	$41.0^{\dagger}$
Multi-Task [Han *et al.*, 2019b]	-	-	**64.5**	$64.2^{\dagger}$	$51.4^{\dagger}$	$57.1^{\dagger}$	$41.0^{\dagger}$	$41.1^{\dagger}$	$41.1^{\dagger}$
BiLSTM	**63.9**	38.9	48.4	55.7	48.3	51.8	24.9	23.8	24.3
UCGraph + BiLSTM	60.0	46.5	52.4^{*}	61.2	52.6	56.6^{*}	32.7	32.9	32.8^{*}
BERT	61.1	54.1	57.4	64.8	52.3	57.9	39.1	39.5	39.9
UCGraph + BERT	62.4	**56.1**	59.1^{*}	**66.1**	**56.9**	**61.2***	**44.5**	**42.3**	**43.4***

Table 3: Performance of different models on TBDense, TDD-Auto and TDD-Man. P, R, and F1 denote Precision (%), Recall (%), and F1-score (%). The shaded lines indicate our method. † denotes our re-implementations. $*$ indicates significance test at a level of p=0.05.

5 Experimental Results

5.1 Overall Performance

Table 3 compares the performance of different models on TDD-Auto and TDD-Man. We also study the performance of different models on the *local* event temporal relation dataset TBDense[7]. From the results, our approach (UCGraph+BERT) achieves the best performance on TDD-Auto and TDD-man, outperforming the other methods by consideration margins (+2.4% in F1 on the average). This has justified the effectiveness of our approach. While, our approach underperforms Deep SSVM and Multi-Task on TBDense, where the reason might be TBDense focuses on local event temporal ordering, which may not fit with our method as the constructed temporal graph is too small for feature learning. We also note that both BiLSTM representations and BERT representations are effective (e.g., +8.5% and +3.5% in F1 on TDD-Man). This suggests that the effectiveness of our approach is independent of the initialized node representations.

5.2 Maintaining Global Consistency

Maintaining global consistency is an important aspect for discourse-level event temporal ordering. For example, if $e_1 \xrightarrow{\text{BEFORE}} e_2$ and $e_2 \xrightarrow{\text{BEFORE}} e_3$, an ideal system should make a globally consistent prediction $e_1 \xrightarrow{\text{BEFORE}} e_3$. Following [Naik *et al.*, 2019], we evaluate global consistency of different models by enumerating every possible triples and check whether the TLINKs are consistent. From the results in Table 4, adding ILP constraints[8] improves both the F1 score and global consistency for feature-based model (e.g, adding ILP constraints leads to a 3-point gain in F1 and a 1.2 point in consistency over SP). However, for neural network based methods (e.g., BiLSTM and BERT), though ILP constraints improves consistency, they generally harm the F1 score. This may show the difficulty in injecting prior knowledge into neural network based models. Our model achieves the highest F1 score and meanwhile can maintain good global consistency,

	TDD-Auto		TDD-Man	
Method	**F1**	**Cons.**	**F1**	**Cons.**
SP	43.2	52.7	22.7	53.5
SP + ILP	46.3 (↑)	53.6 (↑)	23.8 (↑)	54.7 (↑)
BiLSTM	51.8	41.9	24.3	40.0
BiLSTM + ILP	50.9 (↓)	42.4 (↑)	24.4 (↑)	41.1 (↑)
BERT	57.9	53.2	39.9	51.7
BERT + ILP	58.1 (↑)	56.1 (↑)	39.2 (↓)	53.8 (↑)
Deep SSVM	58.8	56.0	41.0	54.2
Multi-Task	57.1	55.8	41.1	54.8
UCGraph	**61.2***	**56.9***	**43.4***	**55.9***

Table 4: Results of maintaining global consistency (Cons.). ↑ denotes a positive impact by adding ILP; ↓ denotes a negative impact. $*$ indicate the significance test with p=0.05.

which justifies the effectiveness of assimilating document-level features via graph representation learning.

6 Ablation and Discussion

We next conduct a series of studies to further explore the effectiveness of our model. The experiments are based on the development set of TDD-Auto, considering that it is much larger than TDD-Man and its annotation quality is also good.

6.1 Impact of the Graph Masking Pre-Training

We compare different edge masking strategies to explore which is the most effective way to learn edge inter-dependencies: 1) One-edge mask. At each step, only one edge is masked, and the goal is to recover the masked edge based on the remaining graph. 2) Random mask. At each step, a portion (ranging from 1% to 50%) of edges are randomly masked. Figure 4 shows the results. Particularly, the random strategy is better than one-edge strategy when a relative small portion (e.g., 5% to 15%) is adopted. A plausible explanation is that the temporal graph is dense, so only masking one edge may prevent the model to learn the underlying patterns. While, when a lot of edges are masked (i.e., when the portion is over 15%), it is also difficult for a model to capture edge inter-dependencies.

[7]Note, on TBDense, we only build local graph and we also discard the VAGUE TLINKs following [Naik *et al.*, 2019].

[8]We use PuLP (https://github.com/coin-or/pulp) to implement ILP algorithm and adopt global constraints.

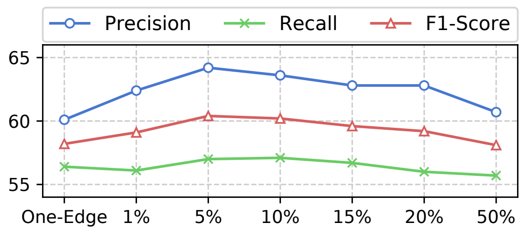


Figure 4: The impact of graph masking pre-training. The experiments are based on the development set of TDD-Auto.

Setting	Method	P	R	F1
Pair-wise	BERT + Ind.	60.0	55.4	57.6
Graph View	Graph + Seq	60.6	56.2	58.3
	Graph + Rand.	62.9	53.8	58.0
	Graph + Logits	62.2	53.8	57.7
	Graph + UC	**64.2**	**57.0**	**60.4**

Table 5: The impact of certainty-first graph completion strategy. The experiments are based on the developing set of TDD-Auto.

6.2 Impact of the Certain-First Strategy

We take a closer look at our certainty-first graph completion strategy, by comparing it with other strategies: 1) BERT+Ind., which makes independent pair-wise predictions based on BERT representations; 2) Graph+Seq., which adopts graph view but predicts edges sequentially (e.g., following the ordering (e_1, e_2), (e_1, e_3), ...); 3) Graph+Rand., which also adopts graph view but randomly predicts an edge at each step; 4) Graph+Logits, which is similar to our model but ranks all of the edge predictions based on their softmax probabilities, and at each step considers the edge having the highest softmax probability as prediction. Table 5 shows the results.

From the results, approaches adopting graph view generally yield better performance than pair-wise methods, which justifies the necessity of graph representation learning. While, the prediction orders affect the results seriously. For example, Graph + Logits even yield negative results than Graph + Rand. adopting random prediction strategy. This implies that softmax probability does not reflect the reliability of prediction. Among all the above strategies, our certainty-first approach achieves the best performance. To find out the reason, we manually examine predictions and their uncertainty values. From the results, for predictions whose uncertainty value are lower than 0.15, 81.2% of them are correct; for predictions whose uncertainties values are higher than 0.5, only 30.1% of them are correct. This implies that our certain-first strategy tend to yield correct predictions at early stage and thus reduce error cascading in graph completion.

6.3 Learning Visualization and Case Study

A salient *cluster* pattern can be observed in our certainty-first graph completion process. Particularly, Figure 5 studies the number of cases where the events evolved in the current edge prediction overlap with that of previous-N predictions, by comparing our method and a random prediction strategy. From the results, in our method, there are 358 cases (28%) and 780 cases (61%) overlap with the previous-1 and -3 predictions, but in the random strategy there are only 48 cases (3%) and 122 cases (9%). This indicate that our method tends to tackle events which have been addressed in previous predictions. Figure 6 visualizes the case of a randomly selected document APW19980227.0494 with down-sampled events, where the cluster pattern can be clearly observed. A plausible explanation of the above phenomenon is that: when an edge is predicted and added into graph, our model gets more information about the nodes (i.e., events). Thus our model becomes more confident to predict edges involving those nodes.

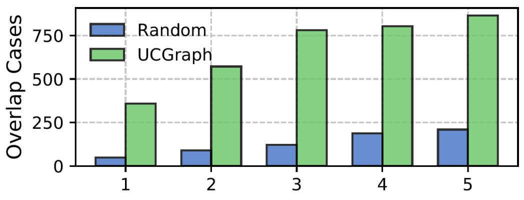


Figure 5: Overlapping cases with previous N predictions.

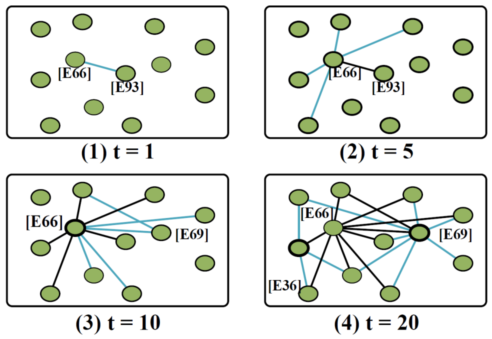


Figure 6: Visualization of the graph completion process. Blue edges indicate the newly predicted edges. For simplicity, we do not distinguish the type/direction of an edge.

7 Conclusion

In conclusion, this paper takes a new graph perspective on the task of discourse-level event temporal ordering, framing it as a graph completion problem. A new model based on graph representation learning and uncertainty modeling is proposed, which can effectively capture the inter-dependencies of temporal relations and assimilate document-level features for reasoning. The experimental results have suggested that our approach not only improves performance but also maintains global consistency. In the future, we seek to apply our method to other discourse-level tasks such as document-level relation extraction and event causality identification.

Acknowledgments

The research work descried in this paper has been supported by the Fundamental Research Funds for the Central Universities 2021RC234. It has also been supported by the National Key R&D Program of China (2019YFB1405200) and the National Nature Science Foundation of China (No. 61976015, 61976016, 61876198 and 61370130).

References

[Allen, 1984] James F. Allen. Towards a general theory of action and time. *Artificial Intelligence*, 23, 1984.

[Bethard *et al.*, 2007] S. Bethard, J. H. Martin, and S. Klingenstein. Timelines from text: Identification of syntactic temporal relations. In *ICSC*, pages 11–18, 2007.

[Cassidy *et al.*, 2014] Taylor Cassidy, Bill McDowell, Nathanael Chambers, and Steven Bethard. An annotation framework for dense event ordering. In *ACL*, 2014.

[Chambers *et al.*, 2014] Nathanael Chambers, Taylor Cassidy, Bill McDowell, and Steven Bethard. Dense event ordering with a multi-pass architecture. *TACL*, 2014.

[Chambers, 2013] Nathanael Chambers. NavyTime: Event and time ordering from raw text. In *SemEval*, 2013.

[Chang and Manning, 2012] Angel X. Chang and Christopher D. Manning. Sutime: A library for recognizing and normalizing time expressions. In *LREC*, 2012.

[Chen *et al.*, 2020] Fenxiao Chen, Yun-Cheng Wang, Bin Wang, and C.-C. Jay Kuo. Graph representation learning: a survey. *TSIP*, 2020.

[Cheng and Miyao, 2017] Fei Cheng and Yusuke Miyao. Classifying temporal relations by bidirectional LSTM over dependency paths. In *ACL*, 2017.

[Devlin *et al.*, 2019] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *ACL*, 2019.

[Do *et al.*, 2012] Quang Do, Wei Lu, and Dan Roth. Joint inference for event timeline construction. In *EMNLP*, 2012.

[Gal and Ghahramani, 2016] Yarin Gal and Zoubin Ghahramani. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *ICML*, 2016.

[Han *et al.*, 2019a] Rujun Han, I-Hung Hsu, Mu Yang, Aram Galstyan, Ralph Weischedel, and Nanyun Peng. Deep structured neural network for event temporal relation extraction. In *CoNLL*, 2019.

[Han *et al.*, 2019b] Rujun Han, Qiang Ning, and Nanyun Peng. Joint event and temporal relation extraction with shared representations and structured prediction. In *EMNLP*, 2019.

[Hochreiter and Schmidhuber, 1997] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 1997.

[Kingma and Ba, 2015] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *ICLR*, 2015.

[Kipf and Welling, 2017] Thomas N. Kipf and Max Welling. Semi-Supervised Classification with Graph Convolutional Networks. In *ICLR*, 2017.

[Liu *et al.*, 2019] Jian Liu, Yubo Chen, Kang Liu, and Jun Zhao. Neural cross-lingual event detection with minimal parallel resources. In *EMNLP*, 2019.

[Marcheggiani and Titov, 2017] Diego Marcheggiani and Ivan Titov. Encoding sentences with graph convolutional networks for semantic role labeling. In *EMNLP*, 2017.

[Naik *et al.*, 2019] Aakanksha Naik, Luke Breitfeller, and Carolyn Rose. TDDiscourse: A dataset for discourse-level temporal ordering of events. In *SIGdial*, 2019.

[Ning *et al.*, 2017] Qiang Ning, Zhili Feng, and Dan Roth. A structured learning approach to temporal relation extraction. In *EMNLP*, 2017.

[Ning *et al.*, 2018] Qiang Ning, Zhili Feng, Hao Wu, and Dan Roth. Joint reasoning for temporal and causal relations. In *ACL*, 2018.

[Pennington *et al.*, 2014] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *EMNLP*, 2014.

[Pustejovsky *et al.*, 2003a] James Pustejovsky, José M. Castaño, Robert Ingria, Roser Sauri, Robert J. Gaizauskas, Andrea Setzer, Graham Katz, and Dragomir R. Radev. Timeml: Robust specification of event and temporal expressions in text. In *New Directions in QA*, 2003.

[Pustejovsky *et al.*, 2003b] James Pustejovsky, Patrick Hanks, Roser Saurí, Andrew See, Rob Gaizauskas, Andrea Setzer, Dragomir Radev, Beth Sundheim, David Day, Lisa Ferro, and Marcia Lazo. The timebank corpus. *Proceedings of Corpus Linguistics*, 2003.

[Reimers *et al.*, 2016] Nils Reimers, Nazanin Dehghani, and Iryna Gurevych. Temporal anchoring of events for the TimeBank corpus. In *ACL*, 2016.

[Schlichtkrull *et al.*, 2018] Michael Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. Modeling relational data with graph convolutional networks. In *ESWC*, 2018.

[UzZaman and Allen, 2010] Naushad UzZaman and James Allen. TRIPS and TRIOS system for TempEval-2: Extracting temporal information from text. In *SemEval*, 2010.

[Veličković *et al.*, 2018] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph Attention Networks. *ICLR*, 2018.

[Verhagen *et al.*, 2007] Marc Verhagen, Robert Gaizauskas, Frank Schilder, Mark Hepple, Graham Katz, and James Pustejovsky. SemEval-2007 task 15: TempEval temporal relation identification. In *SemEval*, 2007.

[Yan *et al.*, 2011] Rui Yan, Liang Kong, Congrui Huang, Xiaojun Wan, Xiaoming Li, and Yan Zhang. Timeline generation through evolutionary trans-temporal summarization. In *EMNLP*, 2011.

[Zhang *et al.*, 2018] Yuhao Zhang, Peng Qi, and Christopher D. Manning. Graph convolution over pruned dependency trees improves relation extraction. In *EMNLP*, 2018.

[Zhou *et al.*, 2019] Ben Zhou, Daniel Khashabi, Qiang Ning, and Dan Roth. “going on a vacation” takes longer than “going for a walk”: A study of temporal commonsense understanding. In *EMNLP*, 2019.