Temporal Relation Extraction with a Graph-Based Deep Biaffine Attention Model

Bo-Ying Su  
University of California, San Diego  
b1su@ucsd.edu

Shang-Ling Hsu  
Hong Kong University of Science and Technology  
shsuaa@connect.ust.hk

Kuan-Yin Lai  
National Taiwan University  
eddy50811@gmail.com

Amarnath Gupta  
University of California, San Diego  
a1gupta@ucsd.edu

Abstract

Temporal information extraction plays a critical role in natural language understanding. Previous systems have incorporated advanced neural language models and have successfully enhanced the accuracy of temporal information extraction tasks. However, these systems have two major shortcomings. First, they fail to make use of the two-sided nature of temporal relations in prediction. Second, they involve non-parallelizable pipelines in inference process that bring little performance gain. To this end, we propose a novel temporal information extraction model based on deep biaffine attention to extract temporal relationships between events in unstructured text efficiently and accurately. Our model is performant because we perform relation extraction tasks directly instead of considering event annotation as a prerequisite of relation extraction. Moreover, our architecture uses Multilayer Perceptrons (MLP) with biaffine attention to predict arcs and relation labels separately, improving relation detecting accuracy by exploiting the two-sided nature of temporal relationships. We experimentally demonstrate that our model achieves state-of-the-art performance in temporal relation extraction.

1 Introduction

Event temporal relation extraction from text is a crucial task in natural language understanding, given its fundamental role in common sense reasoning, information retrieval, and clinical applications, such as disease progression. The extraction of temporal relations with events comprises several sub-tasks, including the identification of event expressions and temporal relations between events and times, as formally defined in SemEval-2016 (Bethard et al., 2016). An example of temporal relation extraction with events is shown in Figure 1, where Dec 19, 2006 is the document creation time, and three event instances occur in the sentence:

"Preliminary results show no sign of an influenza virus in the samples taken from the farm in the Haute-Marne (department),” a statement said.

Figure 1: An illustration of temporal relation extraction with events. For simplicity, only the relations between time expressions and events are shown, but in practice, the relations involving other tokens are set toNONE. Our focus is to extract Event-Event relations in text.

show, taken, and said. According to the annotations, the time span of the event show INCLUDES Dec 19, 2006 and is AFTER said. Besides, said happens BEFORE Dec 19, 2006, while taken happens even BEFORE said. The other relations between time expressions and event instances may be inferred accordingly or marked as VAGUE, such as the one between show and taken.

Early studies in event temporal relation extraction rely on human-engineered features (Wu et al., 2014; Daumé III, 2007; Blitzer et al., 2006), which may experience performance drop upon out-of-domain applications (Dligach et al., 2017). Later, a variety of neural models, such as convolutional neural network (CNNs) (Dligach et al., 2017) and long short-term memory neural networks (LSTMs) (Tourille et al., 2017), outperform hand-crafted feature-based models. Structured learning approaches are also proposed to enhance performance by incorporating more knowledge (Ning et al., 2017; Han et al., 2019). Recent research efforts further explore the power of pre-trained Transformer-based models (Vaswani et al., 2017) on temporal relation extraction by proposing ex-
tensions and variants (Wang et al., 2019; Ning et al., 2019; Yang et al., 2019; Han et al., 2019, 2020). Latest state-of-the-art performance has been achieved by end-to-end, multi-task approaches that conduct both event annotation and temporal relations extraction jointly (Han et al., 2019, 2020; Lin et al., 2020). Nonetheless, even if these systems are able to implement joint information extraction, they suffer from performance issues in the inference speed and accuracy, which may be the result of the computationally expensive decoding process and insufficient fusion of information extracted for classification.

To this end, we propose a Graph-Based Deep Biaffine Attention Model. Leveraging deep biaffine attention mechanisms and a lightweight decoder module while predicting the relations as a unified graph instead of isolated components, our end-to-end, multi-task model speeds up the inference time for around 39% and is able to achieve state-of-the-art performance on benchmark datasets of temporal relation extraction with events.

We summarize our contributions below:

- We propose the novel application of deep biaffine attention mechanisms in information extraction and demonstrate the benefits of separating arc and relation prediction.
- We speedup the decoding process of relation extraction by around 39% with highly parallelizable pipeline without a significance drop in terms of F1 score.
- Our model outperforms the state-of-the-art baselines in temporal relation extraction in benchmark datasets.

2 Problem Specification

We focus on the problem of extracting relations between events from raw text. The goal is to generate a Temporal Relation Graph from unstructured text. A Temporal Relation Graph represents the event-event temporal relationships in a piece of text, with events being the vertices and relations being the edges. Depending on the dataset, the edges in the Temporal Relation Graph may comprise of different set of labels. Our system is end-to-end, which means that event and relation annotations are only given in the training stage. On contrast, during prediction, only raw sentences are given, and the relations are predicted without prior event annotations. Solving such a problem is useful in real-world settings, as neither annotations of events nor relation data is easily accessible.

3 Graph-Based Deep Biaffine Attention Model

In this section, we describe the details of our proposed Graph-Based Deep Biaffine Attention Model.

The input sentences are first tokenized and embedded with a fine-tuned pre-trained BERT model. Then, the sequence of tokens is encoded with a Bidirectional-LSTM layer to create a neural representation of the input. Two separated Biaffine Attention Modules, ARC and REL, will then predict arcs and relation labels between each pair of tokens based on the neural representation respectively. These two pieces of information correspond to the edges and labels in the Temporal Graph. Finally, the decoder combines the most probable arc and temporal relation labels to make predictions of the temporal relations between each pair of tokens. We perform multi-task learning on our model. In other words, the losses of the ARC and REL module are combined to a total loss and are optimized jointly using cross-entropy loss.

Although we do utilize gold event span information during training, event information is not used during inference, which eliminates the requirement of event annotations and furthers the potential of the method to be put into production.
3.1 Preprocessing

In this section, we describe how we preprocess the raw annotations from the dataset.

Since the TimeBank-Dense dataset (TB-Dense) only annotates event relations across sentences that are not further than two sentences away (Cassidy et al., 2014), we apply a sliding window to capture the tokens in every two consecutive sentences and the event-event relations among them.

Then, we take a data augmentation step on the dataset. Consider \( e_1 \subseteq X \) and \( e_2 \subseteq X \) to be two annotated events, each containing one or more tokens from the dataset \( X \), a relation tuple \((x_1, x_2, r)\) containing two tokens \( x_1, x_2 \in X \) and a relation label \( r \) is generated if and only if \((x_1, e_1)\) and \((x_2, e_2)\) and there is TLINK annotation between \( e_1 \) and \( e_2 \) in the dataset. This step effectively produces a dense graph among two event spans if there is a temporal relation between them.

All the inverse relations that could be inferred immediately from the dataset is also added to the training data.

3.2 BERT Fine-tuning

Similar to previous studies that utilizes pre-trained BERT model, we leverage the Masked Language Modeling target with implementation from huggingface (Wolf et al., 2019) to fine-tune the BERT-base model on each training corpus.

3.3 ARC and REL module

Our Graph-Based Deep Biaffine Attention Model makes two predictions for any given text: arc (ARC) and relation (REL). In this section, we describe how ARC and REL modules work.

**ARC** The purpose of the ARC module is to predict the existence of an arc between any pair of tokens. Additionally, the ARC module output also give us information about the direction of the arc, although it is not used. The motivation of the ARC module is to convert the problem of detection the existence of a temporal relationship between tokens to a prediction of a directed edge between a pair of tokens. This effectively gives the model more information about the relations labels than simply telling if there is an arc or not.

For every text input, we create a binary two-dimensional matrix \( ARC_{i,j} \in \{0, 1\}^{n \times n} \), where \( n \) denotes the sequence length, which is exactly the adjacency-list representation of the graph. \( ARC_{i,j} = true \) if only if there is an arc that points from \( x_i \) to \( x_j \), where \( x_i \) denotes the \( i^{th} \) token in the sentence.

For each pair of two-sided temporal relations, such as \{AFTER, BEFORE\} and \{INCLUDE, IS_INCLUDED\}, we arbitrary choose one side to be the source and the other to be the sink. Consequently, if there is a two-sided temporal relationship between a pair of token \( s_i \) and \( s_j \), we have either \( ARC_{i,j} = true \) or \( ARC_{j,i} = true \), but not both.

For those temporal relations that does not have their counterparts, such as \{SIMULTANEOUS\} and \{VAGUE\}, we simply create two edges between them in ARC. For example, if there is a token-relation pair \((t_i, t_j, SIMULTANEOUS)\), the model will set \( ARC_{i,j} = ARC_{j,i} = true \).

A Biaffine attention operator is used to predict the ARC matrix, which we will discuss in the Biaffine Attention section below.

The cross-entropy loss of the ARC module is calculated as follows:

\[
L_{arc} = \text{loss} \left(ARC_{i,j}^{gold}, s_{i,j}^{(arc)} \right) = -\frac{1}{N^2} \sum_{i,j} ARC_{i,j}^{gold} \cdot \log \left(s_{i,j}^{(arc)} \right) + \left(1 - ARC_{i,j}^{gold} \right) \cdot \log \left(1 - s_{i,j}^{(arc)} \right)
\] (1)

Since our model does not have any event information during inference, the ARC module is trained to identify possible arcs among all token pairs. This is accomplished by sampling 50% of token pairs randomly in the input sentence during training.

**REL** The purpose of the REL module is to predict the temporal relation label between any pair of token. Here is a list of temporal relations that is annotated in the two datasets that we used for evaluation.

| TB-Dense | MATRES |
|----------|--------|
| BEFORE   | BEFORE |
| AFTER    | AFTER |
| SIMULTANEOUS | SIMULTANEOUS |
| VAGUE    | VAGUE |
| INCLUDE  | -     |
| IS_INCLUDED | -     |

Table 1: Temporal Relation Set annotated in evaluation datasets.
we use a BiLSTM layer to encode the token embeddings from the BERT model. The encoded sentences are then passed into four independent Multilayer Perceptrons (MLP), which we will describe below.

3.4 Sentence Encoder

Similar to prior temporal relation extraction systems (Han et al., 2019, 2020; Ning et al., 2019), we use a BiLSTM layer to encode the token embeddings. The encoded sentences are then passed into four independent Multilayer Perceptrons (MLP), which we will describe below.

3.5 MLP and Biaffine Attention Layer

The purpose of this Deep Biaffine Attention layer is to predict the ARC matrices and REL tensors which are mentioned earlier. Deep Biaffine Attention Architecture is first used to predict dependency edges (Dozat and Manning, 2016). The predicted edges and labels are used to construct the dependency tree for a sentence, producing superior parsing accuracy and performance. In this work, we modified the Deep Biaffine Attention mechanism to predict directed edges and associated labels for constructing Temporal Relation Graphs.

### Predicting ARC

The goal here is to make a prediction of the ARC matrix. For each sentence encoding output from the BiLSTM layer, we first pass it through two Multilayer Perceptrons (MLP) to reduce its hidden dimension in half, creating two vectors, \( h_i^{(arc-dep)} \) and \( h_j^{(arc-head)} \). The reason of creating two separated vectors is to allow the two MLPs to be trained to distill head and dependent information for the Biaffine Attention Layers from the BiLSTM output, reducing its vector dimensions. Here, we use the term "head" to represent an encoded token being the source of other tokens (pointing outwards), whereas "dep" represents a token being a child of another token.

With \( h_i^{(arc-dep)} \) and \( h_j^{(arc-head)} \), we apply the Biaffine operator to compute the attention of all tokens to every other tokens, creating a two dimensional matrix \( s_{i,j}^{(arc)} \) ∈ \( \mathbb{R}^{m \times m} \), where \( m \) denotes the sequence length:

\[
s_{i,j}^{(arc)} = \text{Biaffine} \left( h_i^{(arc-dep)}, h_j^{(arc-head)} \right)
\]

Biaffine\( (y_1, y_2) = y_1^\top U y_2 + W (y_1 \odot y_2) + b \)

\[
\text{Bilinear} \hspace{1cm} \text{Linear}
\]

Here, \( U, W \) and \( b \) are a \( m \times m \) tensor, \( 1 \times 2m \) matrix and a bias vector, where \( m \) is the sequence length of the given text (same as the length of the BiLSTM output).

The intuition here is to model both the prior probability of a token \( i \) having any dependents \( (W(y_1)) \) and the likelihood of \( i \) receiving a specific dependent \( j \) \( (y_1^\top U y_j) \). In both cases, the higher probability means that we are more confident that there is an arc connected to \( i \) \( s_{i,j}^{(arc)} \) is then served as the prediction of ARC\((i, j)\).

### Predicting REL

The goal of this module is to assign every possible arcs between tokens a temporal relation label. Here, we leverage a Fixed-class Biaffine classifier.

| REL     | INV(REL) |
|---------|----------|
| BEFORE  | AFTER    |
| AFTER   | BEFORE   |
| SIMULTANEOUS | SIMULTANEOUS |
| VAGUE   | VAGUE    |
| INCLUDE | IS_INCLUDED |
| IS_INCLUDED | INCLUDE |

Table 2: Inverse mapping between two Temporal Relation labels.
Similar to predicting the ARC matrix, we feed the sentence encoding output from the BiLSTM layer through two Multilayer Perceptrons (MLP) to reduce its hidden dimension in half, creating two vectors $h^{(arc-dep)}_i$ and $h^{(arc-head)}_j$. Then, the Biaffine Attention operator is used to compute the likelihood $s_{i,j}^{(rel)} \in \mathbb{R}^{m \times m \times n}$ of a token pair $x_i, x_j$ connected by a temporal relation label $k$, where $m$ denotes the sequence length, and $n$ denotes the number of relation labels.

$$s_{i,j}^{(rel)} = \text{Biaffine} \left( h^{(rel-dep)}_i, h^{(rel-head)}_j \right)$$ (5)

$$\text{Biaffine}(y_1, y_2) = y_1^T U y_2 + W (y_1 \odot y_2) + b$$ (6)

Here, $U, W$ and $b$ are a $m \times l \times m$ tensor, $l \times 2m$ matrix and a bias vector where $m$ is the sequence length of the given text (same as the length of the Bi-LSTM output) and $l$ is the number of temporal relation labels.

This construction allows the optimizer to learn the prior probability of a certain label being assigned to any pair of tokens, the likelihood of a head token taking a particular relation and how probable a relation is assigned given a head and a specific dependent. $s_{i,j}^{(rel)}$ is then served as the prediction of $REL(i,j)$.

The Biaffine Attention mechanism allows us to model the temporal relation extraction task directly, instead of introducing additional non-linear activation functions to our model.

### 3.6 Loss function

In our Multi-task model that trains ARC and REL jointly, we compute the loss by simply adding ARC loss with REL loss. $L_{arc-rel} = L_{arc} + L_{rel}$

### 3.7 Decoder

Our Event Module uses the same implementation as (ref??), with two Linear layers and a tanh activation function in between The output of the Event module is an matrix of logits of length 2.

The ARC decoder predicts the arc between each pair of tokens $x_i, x_j$ as

$$ARC_{\text{pred}}(i,j) = \mathbb{1}_{R^+}(s_{i,j}^{(arc)})$$ (7)

where $\mathbb{1}$ is an indicator function.

|            | TB-Dense | MATRES |
|------------|----------|--------|
| # Documents|          |        |
| Train      | 22       | 183    |
| Dev        | 5        | -      |
| Test       | 9        | 20     |
| # Relation Pairs |        |
| Train      | 4032     | 6332   |
| Dev        | 629      | -      |
| Test       | 1427     | 827    |

Table 3: Dataset splits of TB-Dense and MATRES.

The Relation Decoder Module predicts the relation label between each pair of tokens. The first part is predicting the existence and the direction of an arc between two tokens $x_i, x_j$. As $LABEL_{\text{pred}}(i,j)$ and $REL_{\text{pred}}(i,j)$ are defined as follows:

$$LABEL_{\text{pred}}(i,j) = \arg\max_k (s_{i,j,k}^{(rel)})$$ (8)

$$a = ARC_{\text{pred}}(i,j) + ARC_{\text{pred}}(j,i)$$ (9)

$$REL_{\text{pred}}(i,j) = \begin{cases} 0 & \text{if } a = 0 \\ LABEL_{\text{pred}}(i,j) & \text{o.w.} \end{cases}$$ (10)

### 4 Experiment Setup

This section describes the details about the two datasets and evaluation metrics that we used to train and evaluate our system. We use our workstation.

#### 4.1 Dataset

We use TimeML-annotated (Pustejovsky et al., 2004) TimeBank-Dense dataset (TB-Dense) and Multi-Axis Temporal Relations for Start-points (MATRES) dataset to evaluate the performance of our system. Both TB-Dense and MATRES dataset provide event annotations and relations between events. High-quality temporal relation data has always been scarce resource, due to its challenging annotation process. Annotators often overlook some relations that can be implied from text, but hard to recognize at first glance, resulting in a low inter annotator agreement.

#### 4.1.1 TB-Dense

The TB-Dense dataset mitigates such an issue by forcing the annotators to consider each pair of
Table 4: Relations in TB-Dense and MATRES. Acronyms of the relations are underlined.

| Relation          | TB-Dense | MATRES |
|-------------------|----------|--------|
| BEFORE            | 384      | 417    |
| AFTER             | 274      | 266    |
| INCLUDES          | 56       | -      |
| IS_INCLUDED        | 53       | -      |
| SIMULTANEOUS      | 22       | 31     |
| VAGUE             | 638      | 113    |

Table 4: Relations in TB-Dense and MATRES. Acronyms of the relations are underlined.

events that co-exist in sentences that is no father than two sentences away. The result is a dataset that contains approximately 10x more relations per document compared to the original Timebank dataset (Cassidy et al., 2014).

4.1.2 MATRES
Recent work by Ning et al. (2018) further improves the quality of temporal annotations by enforcing a multi-axis annotation scheme, improving the inner-annotator agreements significantly.

We use TB-Dense and MATRES datasets to evaluate our model. The data statistics are provided in Table 4.

4.2 Evaluation Metrics
To be consistent with previous research and the baseline models, we adopt the same standard micro-average scores for TB-Dense dataset. NONE relations are excluded from the calculations, following the convention of Information Extraction tasks.

For MATRES dataset, we also use the same micro-average metric as the baseline models.

Since the predictions from the proposed model is at token-level, and it does not predict event spans, we use the first token in every event as representing the entire event. In other words, all relations that is associated to a event in the gold annotation must exist as relations from/to the first token of the event to be considered correct.

The code for evaluation is provided in fmp/utils/metric.py

5 Baseline Models
We pick the state-of-the-art methods from Feature-Based Systems and Neural Systems to be our baselines.

5.1 Feature-Based Systems Baseline
For the TB-Dense dataset, we use CAEVO (Mirza and Tonelli, 2016) as our feature-based system baseline. CAEVO is a rule-based system with Max-Ent classifier, utilizing linguistic features. It is the best performing feature-based temporal relation extraction system for TB-nse. For the MATRES dataset, we use the end-to-end system CogCompTime as our feature-based system baseline.

5.2 Neural Systems Baseline
We use the system by described by Han et al. (2020) as the baseline of the TB-Dense dataset, as it reports the best end-to-end performance to date. For MATRES, we use the end-to-end system by Han et al. (2019) as our baseline model.

6 Results
Table 5 contains our main results. In this section, we discuss the performance on the two datasets respectively.

6.1 TB-Dense
The proposed Graph Biaffine model outperforms the feature baseline by 11.5% and surpasses the neural baseline by 0.4 in the TB-Dense dataset. When looking at the score breakdown per label in Table 6, the proposed Graph-based Deep Bi-affine model performs much better in predicting INCLUDE and IS_INCLUDED temporal relation labels, while falling short on predicting BEFORE and AFTER. We show that with a dedicated ARC module that detects arcs in the Temporal Graph instead of treating not having a relation merely as NONE label, the relation classifier could be much better at dealing with imbalanced data. In the TB-Dense dataset, BEFORE and AFTER relations account for 46.1% of all temporal relation annotations, while INCLUDE and IS_INCLUDED account for only 7.6% percent in total. Additionally, the F1 score of VAGUE relation improved 7.9%. We speculate this is due to our model being superior in distinguishing whether a relation exist between two tokens or not. SIMULTANEOUS remains to be the only label that is not predicted at all, probably due the the lack of samples in the training set (only 22 according to Table 4).

6.2 MATRES
Our proposed model also outperforms baseline models in MATRES dataset by a margin of 5.7%. Since every event is a single verb in MATRES, the proposed model does not have to ensure all relations that is associated to a event to be present in
### Table 5: Comparison of our Graph-based Biaffine Model with the baselines.

|                | TB-Dense | MATRES |
|----------------|----------|--------|
|                | P   | R   | F1 | P   | R   | F1  |
| CA VE (Mirza and Tonelli, 2016) | 43.8 | 35.7 | 39.4 | 62.7 | 58.7 | 60.6 |
| Han et al., 2019 | 52.6 | 46.5 | 49.4 | 59.0 | 60.2 | 59.6 |
| Han et al., 2020 | **53.4** | **47.9** | **50.5** | **42.6** | **52.6** | **46.5** |
| Graph-based Biaffine | 50.6 | **51.3** | **50.9** | 65.8 | **66.8** | **66.3** |

### Table 6: Comparison of the performance of different labels in the end-to-end settings on TB-Dense dataset.

|                | Baseline | Ours |
|----------------|----------|------|
|                | P   | R   | F1 | P   | R   | F1 |
| B              | 58.6 | 55.7 | 57.1 | 54.3 | 37.4 | 44.2 |
| A              | 67.8 | 51.5 | 58.5 | 57.3 | 35.2 | 43.5 |
| I              | 8.3  | 1.8  | 2.9 | 12.1 | 5.4  | 7.48 |
| II             | -    | -    | - | 15.2 | 7.6  | 10.1 |
| S              | 47.6 | 51.4 | 49.4 | 50.7 | 66.0 | 57.3 |
| Avg            | **53.4** | **47.9** | **50.5** | **50.6** | **51.3** | **50.9** |

7 **Computational Efficiency**

In Table 7, we report the sentence per second performance during inference stage of our model, and compare it to the baseline models. We re-implemented the system by Han et al. (2020) for benchmark. To establish a fair comparison, we removed both the score calculation and standard input/output portion during the evaluation process. Every experiment is performed ten times on the TB-Dense dataset and taking the average. We use a workstation equipped with AMD Ryzen 9 5900x and Nvidia GTX1060 6gb as our test bench. For TB-Dense we trained it for 40 epochs, for MATRES we trained it for 19 epochs.

8 **Ablation Studies**

In this section, we perform ablation tests to demonstrate the effect of each improvement. Quantitative results can be found in Table 8.

8.1 **Advantage of biaffine for label prediction**

We compared the label prediction result of our biaffine model to the modified model using two standard Linear layers. The result showed that the Biaffine Attention Mechanism provide slight improvement over Linear layers. We also observe that our Biaffine Attention model runs faster on average.

8.2 **Advantage of two biaffine modules over one classification for NONE relation**

We use a special ARC module to predict the existence of an arc between two tokens. In this section, we show how having a specialized ARC module in predicting NONE relations to be superior than taking the NONE relation as a label in the REL module.

In this experiment, we completely disable the ARC module and do not take an additional step to filter out NONE relations.

9 **Analysis**

We analyze our experimental results from two aspects: The advantage of separate attention modules and the computational efficiency.

9.1 **Advantage of two biaffine attentions over one classification for NONE relation**

Most prior work uses NONE as a classification label in the temporal relation extraction process. In this work, however, the existence of a relation between two tokens is predicted directly by the ARC module. To show such improvement over traditional methods, we similarly added the NONE label to the REL module and completely disable the ARC module to compare the difference.

We found that without the ARC module, the overall F1 score of relation extraction decreases slightly, primary due to the degraded extraction performance on VAGUE labels. This proves that the detection of relation is better treated as a sepa-
Table 7: Comparison of Relation Extraction Performance.

|                     | TB-Dense | MATRES |
|---------------------|----------|--------|
| Han et al. (2019)   | -        | 125.8  |
| Graph-based Biaffine| 325 sent/s | 386 sent/s |

Table 8: Performance of relation extraction: ablation studies on the major components of our model.

|                     | TB-Dense | MATRES |
|---------------------|----------|--------|
| Biaffine Attention  | 49.3     | 69.1   |
| ARC                 | 49.5     | 68.1   |
| All                 | 50.6     | 65.8   |

9.2 Computational Efficiency

Benchmarks show that in a fixed amount of time, our model processes more sentences than previous studies, while providing comparable accuracy. This is primary due to our model design that removes the event extraction dependency for relation extraction. The prediction of temporal relation labels is performed in parallel with arc prediction, eliminating the need of the non-parallelizable, nested for-loop for extracting event pairs in the previous studies.

10 Conclusions and Future Work

In this paper, we propose a Graph-based Biaffine Attention Model for temporal relation extraction. We show that under end-to-end settings, our model predicts arcs and relation labels in the temporal graph directly, unlike standard two-step process that requires predictions of events before predicting relations. Our model is shown to provide better performance than traditional methods experimentally in F1-score and inference speed on benchmark datasets.

We plan to apply the proposed framework to other temporal reasoning tasks, such as extracting Time-Event and Time-Time relations.

References

Steven Bethard, Guergana Savova, Wei-Te Chen, Leon Derczynski, James Pustejovsky, and Marc Verhagen. 2016. Semeval-2016 task 12: Clinical tempEval. In Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016), pages 1052–1062.

John Blitzer, Ryan McDonald, and Fernando Pereira. 2006. Domain adaptation with structural correspondence learning. In Proceedings of the 2006 conference on empirical methods in natural language processing, pages 120–128.

Taylor Cassidy, Bill McDowell, Nathanel Chambers, and Steven Bethard. 2014. An annotation framework for dense event ordering. Technical report, CARNEGIE-MELLON UNIV PITTSBURGH PA.

Hal Daumé III. 2007. Frustratingly easy domain adaptation. In Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics, pages 256–263.

Dmitriy Dligach, Timothy Miller, Chen Lin, Steven Bethard, and Guergana Savova. 2017. Neural temporal relation extraction. In Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers, pages 746–751.

Timothy Dozat and Christopher D Manning. 2016. Deep biaffine attention for neural dependency parsing. arXiv preprint arXiv:1611.01734.

Rujun Han, Qiang Ning, and Nanyun Peng. 2019. Joint event and temporal relation extraction with shared representations and structured prediction. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 434–444.

Rujun Han, Yichao Zhou, and Nanyun Peng. 2020. Domain knowledge empowered structured neural net for end-to-end event temporal relation extraction. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 5717–5729.

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

- Sentence Encoder: 2-layer BiLSTM, hidden size=400
- ARC module: We use two MLPs of same size, with output feature dimension=300
- REL module: We use two MLPs of same size, with output feature dimension=300
- Both BiLSTM and MLP dropout ratio=0.33
- Optimizer: lr=5e-5, mu=.9, nu=.9, epsilon=1e-12, clip=5.0, decay=.75
- All hyperparameters are tuned manually to optimize the development set in TB-Dense and MATRES respectively

B Dataset Summary

- Please refer to Table 4 for relation label statistics and Table 3 for document statistics

C Reproducibility List

- All code and datasets can be found in the project code base.
- The pre-trained BERT-base-uncased model is provided by Huggingface transformers.
- Hyperparameters can be found in the Hyperparameters section.