Phrase 3 Report on Information Extraction: Temporal Dependency Parsing

Ashim Gupta

1 Task Description

In the task of understanding temporal structure, our goal is to extract temporal relations between events and time expressions in a document. This can be accomplished by using a Temporal Dependency Tree: we can denote events and time expressions with nodes of a tree and the edges can be used to denote the temporal relations between these nodes (Kolomiyets et al., 2012). Additionally, we can have other nodes, called meta-nodes, that can be used to define reference times for other these expressions and events in the document. Let us understand the task with the following example:

Example: *On the other hand, it's turning out to be another very bad financial week for Asia. The financial assistance from the World Bank and the International Monetary Fund are not helping. In the last twenty four hours, the value of the Indonesian stock market has fallen by twelve percent.*

For the above prompt, our aim is to arrange the underlined spans in a tree structure along with a few special nodes called, *meta-nodes*. Every document has a meta node named DCT, which is used to refer to the Document Creation Time. The DCT node connects to the *root* node with the relation 'depend-on'. The task of temporal dependency parsing requires us to connect the three underlined spans (events or time expressions) with the nodes of the tree that provide their relative temporal information. For instance, the event node *helping* will be linked with the DCT as its temporal reference is the time of writing of the document. Similarly, the event *fallen* gets connected to the node denoted by *the last twenty four hours*.

Dataset and Tokenization Issues. We use the temporal dependency tree dataset collected by (Zhang and Xue, 2019), and also use the training, development, and test splits provided by them. The dataset is annotated with five types of temporal relations: *Includes*, *Before*, *Overlap*, and *After*, or *Depend on*. The dataset contains 182 train, 5 development, and 9 test documents. This dataset is based on TimeBank dataset from LDC. The authors Zhang and Xue (2019) have only shared their annotations and not the original text from TimeBank, as the dataset has to be licensed through LDC.

This led to the problem in phase 1. The annotations provided by the authors

are dependent on the tokenization procedure used, which they did not mention in their paper.[1] The reason is that annotations are provided as tuples of sentence index, start word index, end word index etc. They also did not reply to my dataset query. Following this, I manually fixed the tokenization differences in the dataset. For instance, for the example provided, in order to make the annotation compatible, I had to change the tokenization of the second sentence to include the period with the token *helping* as a single token.

2 Models and Experiments

I follow (Zhang and Xue, 2018) and use a neural ranking model for learning a temporal dependency parser. Consider a document D with N number of nodes, and i^{th} node whose gold parent is denoted by x_i. For every node i, we first find the list of possible parents of the node i, and then maximize the likelihood of gold parent x_i:

$$\begin{aligned} L &= -logP(x_i|D) \\ &= -log\frac{exp[s_{i,x_i}]}{\sum_{x'_i} exp[s_{i,x'_i}]} \end{aligned}$$

where s_{i,x'_i} is the score for child-parent pair. The denominator is the sum of scores of all possible candidate parents for the child node i.

For selecting the set of possible parents, I select all event nodes, time expressions and meta-nodes from the start of the document till the next two sentences of the child node i.

2.1 Models

Majority. As a simple majority baseline, for a child node, I simply select the immediately previous time expression, event, or meta-node, along with the most common edge label of 'overlap'.

For the neural models, I use an encoder to obtain representations of child and parent node and then apply a linear classification layer to get the scores (s_{i,x'_i}) for a child and a candidate parent node. We use two different models for this encoder:

AvgGlove. Since I could not submit the model for phase 1, I am using GloVe (Pennington et al., 2014) word embeddings as my first neural baseline and also as a representative of phase 1. In order to embed a node, I first average the GloVe word embeddings of all the tokens from the sentence in which the node is present, and then take its element-wise product with GloVe embedding of the node itself.

[1]Please refer to this github link to see the dataset: `https://github.com/yuchenz/crowdsourced_EN_TDT_corpus/blob/master/data/timebank-dense.expert.tdt`

BERT My main goal of the project is to independently replicate the results reported in the recent EMNLP paper (Ross et al., 2020) for BERT as an encoder. Following (Ross et al., 2020), to represent a child-parent pair, I first construct a pseudo-sentence for the (potential) parent node and a pseudo-sentence for the child node. For instance, for example given in Section 1: for chld node *helping.*, I will use the following as input to BERT encoder-

pseudo-sentence: *helping. The financial assistance from the World Bank and the International Monetary Fund are not helping.*

The pair of pseudo-sentences (one for child, one for candidate parent) are concatenated and separated by the [SEP] token, and then fed into the BERT model. The output from the [CLS] token is then fed to the linear classification layer.

Candidate Parent Selection: As mentioned in the paper, in this dataset, a time expression node can only have meta nodes and other time expression nodes as its parent, i.e. a time expression can not have an event expression as its parent. There is no such restriction for event nodes, i.e. event nodes can have any other node as their parent. This strategy can be used to prune the set of candidate parents for each child node. We will follow this same strategy in our implementation.

BERT with Location Information - Phase 3 Note, that in our vanilla BERT-based model, we do not provide any information about the location of the child and parent nodes in the full document. So, we implement another BERT-based ranking model where we naively augment the representation with location information of the child and candidate parent nodes. For instance, for example, given in Section 1: for child node *helping.*, we will now augment the index of the sentence in the whole document (=1 in this case), along with index of the node in the document (=0, since it is the first node in the document). The pseudo-sentence will now become:

pseudo-sentence: *helping. sent_idx : 1 node_idx : 0 The financial assistance from the World Bank and the International Monetary Fund are not helping.*

BiAffine Parser - Phase 3 As stated in the previous report, and as part of Phase 3, we implement a Graph-based Biaffine parser with BERT as an encoder Dozat and Manning (2017); Kondratyuk and Straka (2019). The original BiAffine parser uses LSTM to contextualize each node, while we use the BERT encoder for this purpose. For more details, please refer to the paper in Dozat and Manning (2017).

2.2 Results and Ablations

Training Details. We use Huggingface transformers (Wolf et al., 2019) repository for our implementation. We fine-tune our models for 20 epochs and

Model	Dev	Test
Majority	19.5/15.0	23.0/18.0
AvgGlove	62.5/53.2	68.1/59.5
BERT-base (this work)	**74.7**/62.4	72.4/61.2
BERT-base + Location	75.9/**63.7**	**75.3**/**63.8**
BERT-BiAffine	65.2/53.9	62.8/55.3
BERT-base (reported)	–/59.0	–/**68.0**

Table 1: **Results**. First number in each cell is the unlabeled accuracy (UA), and second number is the labeled accuracy (LA). Ross et al. (2020) do not report the unlabeled numbers in their paper. + Location denotes the BERT model augmented with location information.

choose the model that performs best on the development set. All other hyperparameters are set to their default values, as suggested by (Devlin et al., 2019).

Evaluation. To evaluate the temporal dependency trees, we follow previous work (Zhang and Xue, 2018) in reporting two metrics: labeled accuracy (LA) and unlabeled accuracy (UA), the only difference being that labeled accuracy considers a prediction correct if both predicted parent of the node and predicted edge label are correct. LA is the fraction of nodes for which both parent and edge labels are predicted correctly. In temporal dependency parsing, since we predict n+1 edges (n=nodes in the document), the precision and recall are redundant and only a single number is used. This is in line with the syntactic dependency literature which uses unlabeled attachment score (UAS) and labeled attachment score (LAS) for evaluation. Correspondingly we also use two such metrics: UA and LA. We want to mention that the paperRoss et al. (2020) only reports labeled score and they call it the F1 score while mentioning that it is the same as accuracy (page 4, footnote 7).

Note that the numbers presented here are slightly different from those presented in the report for phase 2. This is because, in our subsequent explorations, we noticed a great deal of variation in results across different random runs. Therefore, we choose to report the average results of three models trained with three random seeds (seeds = 1,2,3). To see the detailed results across different runs, please see the google sheet [2]

Results. Results are shown in table 1. First, note that contextualized ranking models that use BERT as an encoder outperform non-contextualized ones by large margins, as was observed by Ross et al. (2020). Moreover, for all the systems, unlabeled accuracy scores are greater than labeled scores, which means that all models have a hard time recognizing the edge label. A possible direction

[2] https://docs.google.com/spreadsheets/d/18T23B-I90JwzL0zlqrct2-u1OrapIITICTERc0L-QXI/edit?usp=sharing

could be to further analyze and improve the system performance for edge label prediction.

It is worth noting that the BERT-based BiAffine parser performs worse than the BERT-based ranking model. One possible reason for this could be that the ranking model prunes the possible list of parents for each child node, while the BiAffine parser does not employ such pruning. Among others, this could be one avenue for future exploration.

Location Information is Useful. To study the impact of location information, we augmented our Phase 2 model with the location information of the child and its candidate parents. Results in table 1 show that this simple augmentation of location information improves by more than 2 points on the test set. A possible reason why the location information helps is that it is less probable for a node to connect to farther nodes than the closer ones. Another possible exploration for future work could study more sophisticated ways of augmenting the document level information into our parsing model.

Reproducibility Issue. As can be seen from table 1, results reported in the paper Ross et al. (2020) were significantly better for the test set and were significantly worse for the development set. We also could not reproduce the results reported using author's tensorflow-based implementation[3]. The authors mentioned in our conversation via email that they had also observed similar variation in performance across different runs. One point of discrepancy we observed is that in the paper, they mention averaging results across different random runs, while in our email conversation they mentioned using the best scores from 5 different runs.

3 Conclusion and Future Work

In this report, we implemented a temporal dependency parser with the aim of replicating the results of a recent EMNLP paper Ross et al. (2020). We successfully implemented this model and achieved results competitive with the paper. Additionally, we showed that augmenting position information of nodes in the document improves performance by almost 2 percentage points. We also experimented with a graph-based biaffine parser and found the results to be inferior than a similar ranking model. We suggested that a possible explanation could be that the ranking model uses a candidate pruning strategy while the biaffine model does not. A possible direction of future work could be to employ this pruning strategy with the Biaffine parser.

[3] `https://github.com/bnmin/tdp_ranking`

References

J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. URL `https://www.aclweb.org/anthology/N19-1423`.

T. Dozat and C. D. Manning. Deep biaffine attention for neural dependency parsing, 2017.

O. Kolomiyets, S. Bethard, and M. F. Moens. Extracting narrative timelines as temporal dependency structures. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 88–97, 2012.

D. Kondratyuk and M. Straka. 75 languages, 1 model: Parsing universal dependencies universally. 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 2779–2795, 2019.

J. Pennington, R. Socher, and C. D. Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

H. Ross, J. Cai, and B. Min. Exploring contextualized neural language models for temporal dependency parsing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8548–8553, 2020.

T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, et al. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*, 2019.

Y. Zhang and N. Xue. Neural ranking models for temporal dependency structure parsing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3339–3349, 2018.

Y. Zhang and N. Xue. Acquiring structured temporal representation via crowdsourcing: A feasibility study. In *Proceedings of the Eighth Joint Conference on Lexical and Computational Semantics (* SEM 2019)*, pages 178–185, 2019.