DocTime: A Document-level Temporal Dependency Graph Parser

Puneet Mathur[‡], **Vlad I Morariu**[†], **Verena Kaynig-Fittkau**[†], **Jiuxiang Gu**[†], **Franck Dernoncourt**[†], **Quan Hung Tran**[†], **Ani Nenkova**[†],**Dinesh Manocha**[‡], **Rajiv Jain**[†]
‡ University of Maryland, College Park
† Adobe Research
[‡]{puneetm,dmanocha}@umd.edu
[‡]{morariu,kaynigfi,jigu,dernonco}@adobe.com
[‡]{qtran,nenkova,rajijain}@adobe.com

Abstract

We introduce DocTime - a novel temporal dependency graph (TDG) parser that takes as input a text document and produces a temporal dependency graph. It outperforms previous BERT based solutions by a relative 4-8% on three datasets from modeling the problem as a graph-network with path-prediction loss to incorporate longer range dependencies. This work also demonstrates how the TDG graph can be used to improve the downstream tasks of temporal questions answering and NLI by a relative 4-10% with a new framework that incorporates the temporal dependency graph into the self-attention layer of Transformer models (Time-transformer). Finally, we develop and evaluate on a new temporal dependency graph dataset for the domain of contractual documents, which has not been previously explored in this setting.

1 Introduction

Understanding the temporal relations between events mentioned in a document is an important natural language task with applications in downstream tasks such as timeline creation (Leeuwenberg and Moens, 2018), time-aware summarization (Noh et al., 2020), temporal question-answering (Ning et al., 2020a), and temporal information extraction (Leeuwenberg and Moens, 2019). This area of research remains important yet challenging due to several limitations such as confounded modalities (eg. events that are certain to happen vs the ones that might happen), event ambiguity (eg. agreeing to terms of a contract vs signing a contract) and need for complete annotation of all event pairs for precise temporal localization (Yao et al., 2020a).

Early work densely annotated all pairs of events to address this problem (Cassidy et al., 2014), but was limited to short passages or adjacent sentences due to the $\binom{n}{2}$ complexity of the task, especially for long documents. Recently this problem formulation was significantly simplified using temporal dependency trees (TDT) (Zhang and Xue, 2019) and temporal dependency graphs (TDG) (Yao et al., 2020a) by only capturing the reference TIMEX or event to build a dependency graph to capture this information. This enabled the development of temporal dependency parsers (Zhang and Xue, 2018a; Ross et al., 2020a) to infer temporal relationships more robustly and efficiently.

We introduce DocTime - a state-of-the-art temporal dependency parser that parses document-level text to produce temporal dependency graphs. Unlike previous approaches using contextual features such as BERT(Ross et al., 2020b), our model utilizes a graph network and a novel path prediction loss to reason over long-range multi-hop dependencies while maintaining global consistency of temporal ordering of inter-dependent events.

To validate the utility of DocTime and our generated temporal dependency graph, we go one step further than prior work and explore the question of whether temporal dependency graphs are useful for downstream tasks by introducing Time-Transformer. It is a framework to incorporate temporal dependency graphs into existing transformer-based architectures without retraining from scratch. We demonstrate the usefulness of our proposed Time-Transformer on temporal NLI (Vashishtha et al., 2020) and time-sensitive question answering (Chen et al., 2021) tasks.

Prior work on temporal relationship extraction and temporal dependency parsing have been mostly limited to news (Zhang and Xue, 2019; Yao et al., 2020a; Pustejovsky et al., 2003a), narrative stories (Zhang and Xue, 2018b; Kolomiyets et al., 2012) or clinical notes (Bethard et al., 2016). In addition to experimenting with existing temporal dependency parsing datasets, we introduce a dataset for temporal dependency graphs in a new domain - contractual documents, where temporal reasoning over events has real world legal and monetary implications for users.

Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 993 - 1009

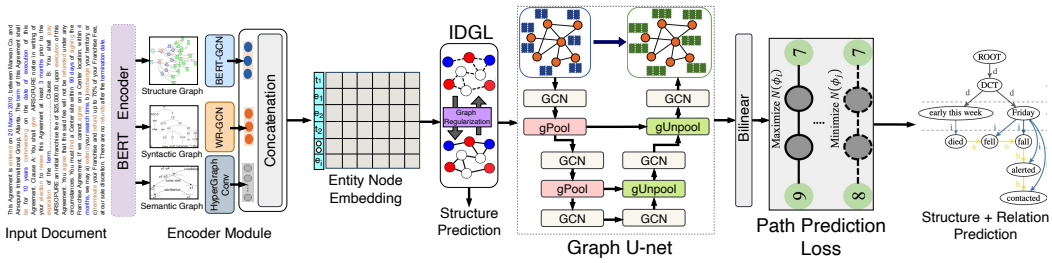

Figure 1: `DocTime` encodes rich token level embeddings from input document using structural, syntactic, and semantic graphs through BERT-GCN, WR-GCN and HyperGraph Conv layers, respectively. Token-level features are concatenated and passed through Iterative Deep Graph Learning (IDGL) to learn a noisy dependency structure over the TIMEX and Event entities. Graph U-net allows the model to incorporate longer range dependencies for predicting the final temporal dependency graph structure and relationships. The model is trained with a novel auxiliary path prediction loss to learn multi-hop connections in TDG.

Our main contributions include:

- A novel document-level temporal dependency parser (`DocTime`) that predicts the temporal dependency graph from text in an end-to-end manner with a novel path prediction loss, which outperforms the current SOTA by a relative 4-8% on three datasets.
- `Time-Transformer`, a novel framework to incorporate Temporal Dependency Graphs into transformer models for downstream tasks without needing to retrain from scratch. Results on natural language inference and question answering with a new self-attention module show a relative 4%-10% improvement.
- Development of new document-level (>1500 words) TDG dataset in the domain of contractual documents (ContractTDG[1]).

2 Related Work

Temporal Dependency Parsing: Previous work has been devoted to pairwise classification of relations between events and time expressions, notably TimeBank (Pustejovsky et al., 2003b) and its extensions like Cassidy et al. (2014) annotated all relations. Pair-wise annotation have multiple problems including polynomial square complexity, global inconsistencies in predictions due to relation transitivity and forced annotation of vague relations (Ning et al., 2018). Prior work focuses on extracting temporal relations between event pairs in the same sentence or adjacent sentences (Goyal and Durrett, 2019; Ning et al., 2019a; Han et al., 2019a,c,b, 2020; Ballesteros et al., 2020; Zhao et al., 2020). TIMERS Mathur et al. (2021a) presented temporal relation extraction in long document.

Temporal Dependency Parsing (TDP): Temporal dependency trees were first proposed by Kolomiyets et al. (2012). (Zhang and Xue, 2018b) provided the the earliest TDT corpus on news data and narrative stories, (Zhang and Xue, 2019) released the first English TDT corpus. Yao et al. (2020a) relaxed the assumption of single reference edge in dependency trees to form the improved TDG. (Zhang and Xue, 2018a) built an end-to-end neural temporal dependency parser using BiLSTM and Ross et al. (2020b) improved it further incorporating BERT. Our approach improves by modeling complex dependencies and introduces a new resource for TDG in contracts.

Linguistically-aware Transformers: Recent works have investigated using linguistic features as a prior for Transformer models. Syntax-bert (Bai et al., 2021a) uses syntactic and constituency dependency on NLI and GLUE benchmarks. Coref-BERT Coreference-Informed Transformer (Liu et al., 2021) performs coreference-aware dialogue summarization. Temporal reasoning about event ordering can find applications in many tasks such as summarization (Noh et al., 2020), question answering (Chen et al., 2021; Ning et al., 2020b; Jin et al., 2020), commonsense reasoning (Qin et al., 2021), and natural language inference (Vashishtha et al., 2020). We propose to use TDG as priors to Transformer models to make them temporally-aware for use in downstream tasks.

3 `DocTime`: Document TDG Parsing

Task Formulation: Let document D be defined as a sequence of n tokens $[x_1, \cdots, x_n]$. The entire document can be seen as sequence of m sentences $[s_1, \cdots, s_m]$. Each document has a set of p events $E = [e_1, \cdots, e_p]$ and q timexes $T = [t_1, \cdots, t_q]$, where $p, q \leq n$. The creation date of the document is represented by timestamp t_{DCT}. Yao

[1] https://github.com/contractTDG/ContractTDG_Dataset

et al. (2020a) defines a temporal dependency graph (TDG) where each timex node always has a reference timex, which is the most specific narrative time related to the event (Pustejovsky and Stubbs, 2011). If such a narrative time is not available, the timex should be anchored to the DCT. An event node can either have a reference timex or be connected to a reference event, which is an event that provides the most specific temporal location. The task of temporal dependency graph parsing of a text document D results in a dependency graph $G = (C, V)$, where C represents the set of all events, timexes and the document creation date (DCT). V is the set of all edges in the graph, where each edge represents a temporal relationship $\Re$ between corresponding entity node pair $V = \{(t_i, t_j), (e_i, e_j), (e_i, t_j)\} \forall i, j \in C$.

Model Overview: Figure 1 shows an overview of our network architecture for temporal dependency parsing. We first extract token level BERT features from the input document, which are then enriched by three graph networks that encode structural, syntactic, and semantic relationships. This is followed by Iterative Deep Graph Learning over the TIMEX and Event entities to learn an initial dependency structure. This is passed through a Graph U-net to allow the model to incorporate longer range dependencies before predicting the final temporal dependency graph and relationships. The model is also trained with a novel auxiliary path prediction loss.

3.1 Feature Encoding

We leverage the pre-trained BERT language model to obtain the embeddings for each token as follows: $w_1, w_2, \cdots, w_n = \text{BERT}([x_1, x_2, \cdots, x_n])$, where w_i is the embedding of the token x_i. As document sequence lengths can be larger than 512, we use a *sliding window* encoding technique to encode whole documents. We average the embeddings of overlapping tokens of different windows to obtain the final representations. These token representations are enriched with slightly enhanced variants of the structural (G_{str}), syntactic (G_{syn}) and semantic (G_{sem}) graphs utilized by (Mathur et al., 2021b) for document-level temporal relationship extraction. The key differences are the use of BERT-GCN (Lin et al., 2021) to combine contextual and structural graph features, the addition of co-reference relationships to the syntactic graph, and the use of a hypergraph convolution (Bai et al., 2021b) to allow for token level features in the semantic graph . All aspects of these features and the changes are presented in Appendix B.

3.2 Temporal Dependency Prediction

We combine the learned representation for each entity node (timex, event, DCT) by concatenating the node embeddings learned from structural, syntactical and semantic graphs to obtain a D-dimensional feature vector for each of z entities in the document given by $\mathbf{F}(w_i) = g_i^{str} \oplus g_i^{syn} \oplus g_i^{sem}$, where $\oplus$ represents concatenation. We retain only the enriched node embeddings for each word. We then utilize Iterative Deep Graph Learning (IDGL)[2] (Chen et al., 2020) to dynamically learn an initial dependency graph structure from the combined node embeddings. Given a noisy graph input feature matrix $\mathbf{F} \in R^{l*D}$, IDGL produces an implicitly learned graph structure $G^* = \{A^*, \mathbf{F}, \mathcal{F}_l\}$ with a jointly refined corresponding graph node embeddings $\mathbf{F}'$ with adjacency matrix A^* by optimizing with respect to downstream link prediction task $\mathcal{F}_l$ between entity nodes.

3.2.1 Graph U-net For Higher Level Features

The Graph U-net (Gao and Ji, 2019) is a U-shaped graph encoder-decoder architecture containing two down-sampling graph pooling (gPool) layers and two up-sampling graph unpooling (gUnpool) layers with skip connections. gPool layers reduce the size of the graph to encode higher-order features, while the gUnpool layer restores the graph into its higher resolution structure, thereby promoting information exchange between entity pairs through an enlarged receptive field. Each graph pooling and unpooling layer is followed by a GCN layer to implicitly capture the topological information in the input graph. Taking the dynamically learned graph structure G^*, a graph embedding layer converts input node features **F'** into low-dimensional representations that are then passed through a graph U-net encoder-decoder $\mho$ to acquire entity-level relation matrix $\mathbf{Y} = \mho(\mathbf{F'})$, $\mathbf{Y} \in R^{l*l*D'}$.

3.2.2 Temporal Dependency Link Prediction and Relation Classification

Given entity adjacency matrix A^* and entity-level relation matrix $\mathbf{Y}$, we use a bilinear function to map them to link and relation probabilities $\mathbf{Z}_l$ and $\mathbf{Z}_r$, respectively. Formally, we have $\mathbf{Z}_l = \sigma(\mathbf{Y}W_l\mathbf{Y} + b_l)$ and $\mathbf{Z}_r = \sigma(A^*W_rA^* + b_r)$, where

[2]Implementation: https://github.com/graph4ai/graph4nlp

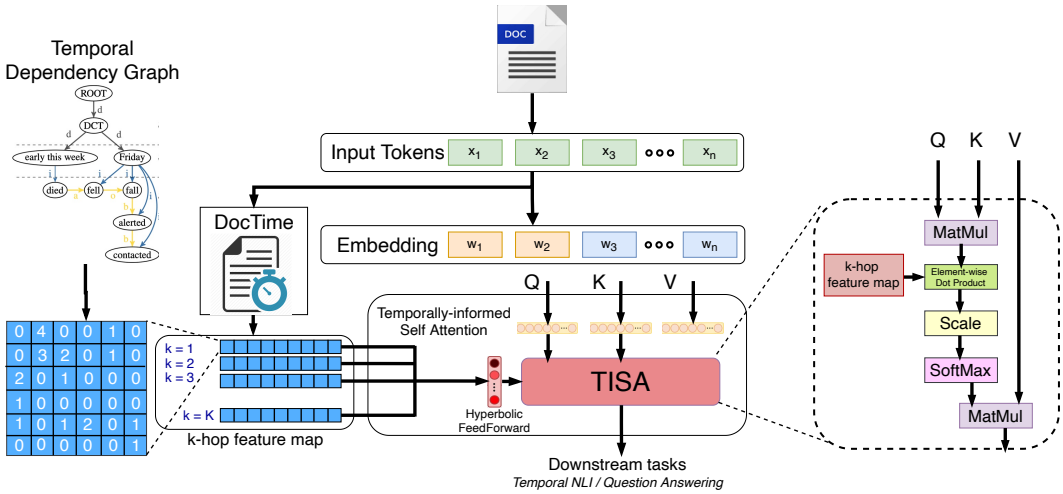

Figure 2: `Time-Transformer` is a variant of pre-trained Transformer models that augments temporal knowledge into the self-attention layer during fine-tuning of the Transformer model on different downstream tasks. Input text is converted into a temporal dependency graph using `DocTime` parser. The graph is then converted into a set of masks that encodes the temporal relationship between each token (i.e. After, Before) using the novel Temporally- informed Self-Attention (TISA). TISA creates K masks to represent the (k)-hop distance between two nodes in TDG for aggregating information across longer ranges in the input. TISA uses hyperbolic feed-forward layer to learn the mask weights.

$W_l, W_r, b_l, b_r \in R^{D'*D'}$ represent learnable parameters. This is followed by a Softmax layer for link prediction and relations classification.

3.3 Training `DocTime`

Path Reconstruction Loss: In a document-level temporal parsing setup, the majority of node pairs may not have any ground truth link or temporal relation. Graph representation learning methods universally model relations between all entity pairs regardless of whether the entity pair has any relationship, leading to dispersion of attention in learning most non-existent edge connections. We propose path reconstruction loss L_{path}, which forces the model to pay more attention to learn entity pairs with relationships rather than ones without relationships. Equation 1 gives the cross entropy loss over all direct edge connection between all pairs of entities, where r_j^i indicates the relation between the entity pair and $P(r_j^i)$ is probability of relation label r. Path reconstruction loss L_{path} modifies the cross entropy loss L_{ce} function as shown in Equation 2 by sampling all n^2 entity pairs and maximizing the probability of the shortest dependency path $\mathcal{N}(\phi)$ between the entity pair nodes. Finally, the path reconstruction loss and the existing classification loss are added as the training objective for `DocTime`, given by $L = L_{path} + L_{ce}$.

$$L_{ce} = -\frac{1}{\sum_{i=0}^{l} N_i} \sum_{i=1}^{l} \sum_{j=1}^{N_i} \{r_j^i \log P(r_j^i) + (1 - r_j^i) \log(1 - P(r_j^i))\} \quad (1)$$

$$L_{path} = -\frac{1}{\sum_{i=0}^{l} N_i} \sum_{i=1}^{l} \sum_{j=1}^{N_i} \{r_j^i \log \mathcal{N}(\phi_i) + (1 - r_j^i) \log(1 - \mathcal{N}(\phi_i))\} \quad (2)$$

Multi-task Training: Dependency link prediction and entity-level relation classification are correlated tasks and reinforce each other. We use multi-task training to optimize both tasks simultaneously using the path prediction cross entropy loss. The final optimization uses a weighted sum of the dependency link prediction loss and entity-level relation classification loss $L = \lambda L_l + (1 - \lambda) L_r$, where the weighting factor λ is a hyperparameter.

4 Time-Transformer

We would also like to understand our temporal dependency parsing can be useful for downstream tasks requiring temporal reasoning. Here we introduce the Time-Transformer, which allow a TDG generated by `DocTime` to be combined with state-of-the-art transformer models for temporal tasks. The Time-Transformer augments the flow of information in a Transformer network via a temporally-informed self-attention mechanism. We first for-

mulate the Time-Transformer architecture in §4 and then construct of temporally-informed attention layers in §4.

Architecture: Time-Transformer was motivated by recent work incorporating syntax (Bai et al., 2021a) or co-reference graphs(Liu et al., 2021) into the transformer architecture to improve downstream tasks. In each case, these approaches encode additional knowledge from the sparse graphs as a masked self attention layer into the transformer. Figure 2 shows the architecture of Time-Transformer incorporating temporal knowledge into the self-attention layer during fine-tuning of the Transformer model. Input text is converted into a temporal dependency graph using `DocTime` parser. The graph is then converted into a set of masks that encodes the temporal relationship between each entity (i.e. After) explained in more detail in the next section: Temporally-informed Self-Attention. The input embedding (token+positional+attention masks) is passed through the Time-Transformer model which modifies the self-attention layer of the standard Transformer architecture with a *temporally-informed self-attention* layer to be fine-tuned on downstream tasks.

TISA: Temporally-informed Self-Attention : The TDG produced by `DocTime` is sparse and to effectively utilize the graph extracted by the temporal dependency parser for longer range temporal relationships, we utilize K self-attention layers that encode the temporal relationship if traversing K hops in the TDG as shown in 2. More formally starting from node A, the minimum number of hops (k) required to reach another node B can be regarded as k-hop distance between A and B, written as k-hop(A, B). We create K masks to represent the (k)-hop distance between two nodes to allow the model to aggregate information across longer ranges in the TDG. Specifically, a mask $M \in \{0, 1, 2, \cdots, r\}^{n \times n}$ denotes if there is a relation between entity i and j, and n is the number of tokens in the input text. The value of the mask is the relationship type for i and j. It is found by inferring the relationship using Allen's interval algebra (Allen, 1983) and is set to 0 if there is no relationship or set to "Overlap" if there is a conflict. We adopt a soft-mask learning strategy to enable the self-attention layer to re-weight the importance of each mask and avoid the problem of vanishing gradient. A hyperbolic feed-forward layer is used to learn the mask weights as research has shown it can avoid distortion of the feature space in graph representations (Ganea et al., 2018). The value of K is a hyperparameter that can be customized according to the nature of input dependency graph.

Training `Time-Transformer`: For each dataset, we optimize the hyper-parameters of `Time-Transformer` through grid search on the validation data. In all our experiments, we limit the maximum value of k-hop to 15. Detailed settings can be found in the appendix.

5 Experiment

5.1 Temporal Graph Parsing Datasets

We train and evaluate `DocTime` on three datasets. First is the **Temporal Dependency Graphs (TDG) dataset** (Yao et al., 2020a) made up of 500 Wikinews articles annotated with document-level temporal dependency graphs. Second is the **Temporal Dependency Trees (TDT) dataset** Zhang and Xue (2019) made from 183 documents derived from TimeBank (Pustejovsky et al., 2003a) annotated with a temporal dependency tree structure. The third dataset we created as part of this paper and is describe in more detail below.

Contract-TDG: Understanding the temporal relationship of events in contracts is an important business problem, where understanding event timelines can have legal and monetary consequences. Previous work on temporal relationships has largely focused on clinical, news or narrative text , whereas to the best of our knowledge the contractual domain has not been explored for this problem. To construct this dataset, we used 100 contracts from the Atticus contracts dataset[3] (Hendrycks et al., 2021), which were sourced from public domain SEC contracts. Due to the multi-page length of these documents, we limited the annotations to the first 1500 words. We did not include definition sections, since they did not contain many events of interest for this task. The documents have a 70-10-20 split for training, validation, and testing.

To obtain the TDG annotations required for our task, we followed the 5 steps procedure outlined by the original TDG dataset in (Yao et al., 2020b): (i) TIMEX Identification (TE), (ii) Identifying reference times for TE, (iii) Event identification, (iv) Identifying reference times for events, (v) Identifying reference events for events. Document Creation

[3]https://www.atticusprojectai.org/cuad

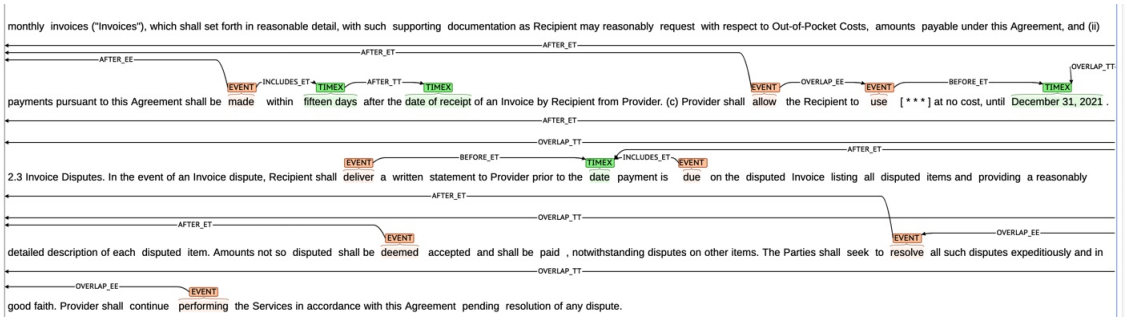

Figure 3: Example of a temporal dependency graph from ContractTDG dataset annotated using Brat Tool.

Dataset	Docs	Timex	Events	Rels
TimeBank (Pustejovsky et al., 2003b)	183	1,414	7,935	6,148
TB-Dense (Cassidy et al., 2014)	36	289	1,729	12,715
MATRES (Ning et al., 2019b)	275	-	1,790	13,577
TDT-Crd (Zhang and Xue, 2019)	183	1,414	2,691	4,105
TDG (Yao et al., 2020a)	500	2,485	14,974	28,350
Contract-TDG) (Ours)	**100**	2354	11,752	12,909

Table 1: Comparison of ContractTDG data statistics to other temporal relation datasets. ContractTDG has fewer documents but comparable number of TIMEX/Events/relations.

Task	TDG (F1)	Contract TDG (F1)
1: TIMEX ID	0.96	0.93
2: TIMEX RT	0.89	0.81
3: Event ID	0.79	0.76
4: RT ID (U)	0.67	0.83
4: RT ID (L)	0.61	0.75
5: RE ID (U)	0.59	0.85
5: RE ID (L)	0.52	0.79

Table 2: Inter-Annotator Agreement (IAA) for the Contract-TDG and TDG dataset. U = structure, L = structure + labels

Times (DCT) were provided as effective dates in the ATTICUS corpus.

Similar to (Yao et al., 2020b) for tasks 1 (TE) and 3 (Event ID), we used the Mechanical Turk platform to obtain two annotations to validate text spans of noisy TIMEXes extracted by HeidelTime software[4] (Strötgen and Gertz, 2013) and verbs that were possible events. Disagreements were resolved by an expert annotator. However, for the reference tasks, we decided against using Mechanical Turk due to the difficulty and length of the contracts as well as the lower agreement faced by the original TDG system for the last two tasks. We instead used the BRAT annotation tool[5] (Stenetorp et al., 2012) with an expert annotator for tasks 2,4, and 5, following the (Yao et al., 2020b) guidelines . ContractTDG is annotated for four temporal relations - after, before, overlaps, and includes.

Table 1 compares the data statistics of the ContractTDG to previous temporal relationship and temporal dependency corpora. Even though this dataset has many fewer documents than the TDG dataset, it has a large number of TIMEX, Events, and Temporal relationships due to the document length. Table 2 reports the F1 IAA metrics for ContractTDG dataset to directly compare to the original TDG dataset. For Tasks 1 and 3 we report IAA F1 for the two crowd sourced worker annotations and for the relationship tagging tasks (2,4,5), we report IAA metrics calculated on the test postion (20% of the data) that was reviewed by two experts. The agreement is slightly lower for the TIMEX/Event identification tasks but higher for the three relationship tasks. We evaluate `DocTime` for dependency structure as well as structure+relation prediction for both development and test splits.

5.2 `Time-Transformer` Experiments for Downstream Tasks

We adopt `Time-Transformer` on BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019a), BigBird (Zaheer et al., 2020a) and FiD (Izacard and Grave, 2021) for evaluation on two downstream tasks in §6.2. We utilized the official checkpoint for each pre-trained language model as provided by respective authors. First, we test `Time-BERT` and `Time-RoBERTa` on Temporal NLI dataset, which consists of 5 sub-datasets (Vashishtha et al., 2020) to study the effect of temporal reasoning for predicting event ordering and duration. Second, we run experiments on the TimeQA dataset (Chen et al., 2021) to evaluate the performance of `Time-BigBird` and `Time-FiD` for the long-document question-answering task. We report Exact Match (EM) and F1 scores as evaluation metrics on dev and test sets of easy and hard versions.

[4]https://github.com/HeidelTime/heideltime
[5]https://brat.nlplab.org/

	System	TD-Trees				TD-Graphs				ContractTDG			
		Structure-only		Structure+Relation		Structure-only		Structure+Relation		Structure-only		Structure+Relation	
		Dev	Test	Dev	Test	Dev	Test	Dev	Test	Dev	Test	Dev	Test
Baselines	Majority Baseline	0.43	0.42	0.15	0.18	0.62	0.68	0.41	0.51	0.36	0.35	0.36	0.33
	Logistic Regression Baseline (Zhang and Xue, 2018a)	0.64	0.70	0.26	0.29	0.62	0.69	0.49	0.58	0.42	0.39	0.45	0.38
	Neural Ranking Parser (BiLSTM) (Zhang and Xue, 2018a)	0.75	0.79	0.53	0.60	0.69	0.79	0.55	0.66	0.49	0.46	0.52	0.48
	BERT Ranking Parser (Ross et al., 2020b)	0.77	0.83	0.59	0.68	0.71	0.80	0.62	0.71	0.67	0.65	0.62	0.61
	DocTime (ours)	**0.85***	**0.86***	**0.66***	**0.72***	**0.74***	**0.85***	**0.69***	**0.77***	**0.70***	**0.69***	**0.68***	**0.64***
Ablation	DocTime w\o Graph U-net	0.83	0.84	0.63	0.70	0.71	0.82	0.67	0.75	0.68	0.63	0.66	0.62
	DocTime w\o Structure Graph	0.81	0.80	0.62	0.65	0.67	0.72	0.65	0.73	0.67	0.63	0.64	0.60
	DocTime w\o Syntactic Graph	0.80	0.82	0.62	0.66	0.65	0.73	0.62	0.69	0.64	0.61	0.62	0.59
	DocTime w\o Semantic Graph	0.76	0.78	0.55	0.65	0.62	0.70	0.60	0.67	0.59	0.57	0.59	0.57
	DocTime w\ Graph Prediction	0.72	0.64	0.49	0.55	0.57	0.65	0.57	0.58	0.59	0.53	0.55	0.54
	DocTime w\ Pairwise Link Prediction	0.82	0.83	0.63	0.69	0.72	0.83	0.66	0.73	0.65	0.60	0.62	0.60
	DocTime w\ Path Prediction Loss	0.85	0.86	0.66	0.72	0.74	0.85	0.69	0.77	0.70	0.69	0.68	0.64

Table 3: Results comparing performance of DocTime with baselines and ablative components on TDT, TDG, ContractTDG datasets. We majority and logistic regression baselines from (Zhang and Xue, 2018a). * indicates statistical significance over BERT Ranking Parser (Ross et al., 2020b) ($p \leq 0.005$) under Wilcoxon's Signed Rank test. Darker green represents better F1 performance on ablation studies. Bold denotes the best performing model. DocTime improves substantially over all datasets for both dependency structure and structure+relation prediction tasks. The ablation shows that semantic graph features prove to be most beneficial. Our proposed path prediction loss is critical for state-of-the-art performance of DocTime model.

6 Results and Analysis

6.1 Temporal Graph Parsing

Performance of DocTime w.r.t. baselines: Table 3 compares the performance of DocTime against other baseline methods on TDT, TDG and ContractTDG. We also provide a majority baseline ContractTDG to evaluate whether the methods work better than a random label assignment as implemented in (Yao et al., 2020a). We also include the two current SOTA approaches for temporal dependency parsing: The BiLSTM attention-based Neural Ranking Parser proposed by (Zhang and Xue, 2018a) [6] and the BERT Ranking Parser (Ross et al., 2020b) on each dataset . We also report results for a logistic regression baseline proposed by (Zhang and Xue, 2018a). Results in Table 3 show that DocTime outperforms both Neural and BERT Ranking Parser by a significant margin on the TDT (2-4%) TDG (5-6%) and ContractTDG (3-4%) datasets. We believe its primarily because they formulate temporal dependency parsing as a ranking task designed to select the best reference event/timex for each node. However, TDG parsing requires the model to be able to reason over multiple dependencies originating from each node while maintaining global consistency of temporal ordering of inter-dependent events. We perform experiments for dependency structure prediction and structure+relation prediction and find that predicting labeled dependency edges is a much more challenging task across all datasets. DocTime achieves state-of-the-art performance on all three datasets (see **bold**), and shows that it can successfully handle document-level long-range dependencies in the challenging ContractTDG dataset from the 6-12% relative improvement over the BERT based ranking parser. A more detailed analysis of performance per temporal relationship type can be found in the Appendix, where largest gains are seen for event-event pairs.

[6] Used: http://github.com/yuchenz/tdp_ranking

Model	UDS-duration	UDS-order	TempEval3	TimeBank-Dense	RED
Majority	50.00	54.52	54.57	50.54	52.51
NBOW (Iyyer et al., 2015)	82.54	54.52	54.57	50.54	52.51
Infersent (Conneau et al., 2017)	92.65	73.22	62.20	68.29	63.47
RoBERTa (Liu et al., 2019b)	94.51	80.17	54.57	94.60	80.59
Time-RoBERTa (E)	95.78	82.03	60.66	95.45	82.10
Time-BERT	96.01	82.97	61.32	96.08	82.15
Time-RoBERTa	**96.67**	**82.98**	**62.50**	**96.33**	**82.50**

Table 4: Accuracy comparison on the Temporal NLI dataset test set. Time-RoBERTa fine-tuned by utlizing temporal dependencies extract from DocTime model pre-trained on TDG dataset outperform all baselines provided by (Vashishtha et al., 2020)(see **bold**).

Ablation Study of DocTime: To assess the contribution of structure and syntactic and semantic graph features, we performed ablation experiments as reported in Table 3 highlighted in red. We also analyzed the effect of different types of training loss. We observe that removing the semantic graph consistently degrades performance, indicating the need for hypergraph learning over temporal arguments and RST features to capture document-level discourse relations. We see that removing structure graph reduced the performance to below the BERT Ranking Parser, as DocTime leverages BERT's contextual learning through a structural graph. Syntactic graph adds incremental value to DocTime due to its relational learning of syntactic dependencies within each sentence through relational GCN. We evaluated the model performance in case all edges of the TDG are used for one forward pass and call it "Graph Prediction". Training the model by evaluating a single edge in one pass (similar to temporal relation prediction in (Pustejovsky et al., 2003b) is referred to as "Pairwise Prediction". We explore the impact of different training

Model	Easy-mode				Hard-mode			
	Dev		Test		Dev		Test	
	EM	F1	EM	F1	EM	F1	EM	F1
FT on TimeQA								
BigBird (Zaheer et al., 2020b)	16.4	27.5	16.3	27.1	11.4	20.6	11.9	20.3
Time-BigBird (E)	15.5	25.0	14.1	25.5	9.6	15.6	9.3	18.5
Time-BigBird	**18.9**	**29.5**	**18.9**	**29.5**	**13.0**	**22.5**	**13.0**	**22.8**
FiD (Izacard and Grave, 2020)	15.9	27.1	15.7	28.0	10.7	19.1	10.3	19.7
Time-FiD (E)	13.8	25.2	12.1	25.6	8.9	17.3	8.8	17.6
Time-FiD	**17.5**	**29.3**	**18.1**	**30.3**	**12.5**	**22.2**	**12.5**	**21.5**
FT on TriviaQA								
BigBird (Zaheer et al., 2020b)	33.4	42.5	33.7	43.0	27.7	35.9	27.7	36.2
Time-BigBird (E)	31.3	40.4	32.3	41.8	25.9	33.6	25.8	35.5
Time-BigBird	**35.0**	**44.8**	**35.1**	**45.5**	**29.2**	**36.6**	**29.2**	**38.0**
FT on NQ + TimeQA								
FiD (Izacard and Grave, 2020)	59.5	66.9	60.5	67.9	45.3	54.3	46.8	54.6
Temp-FiD (E)	57.9	65.6	58.5	65.2	41.1	52.6	44.5	52.8
Time-FiD	**61.3**	**68.2**	**62.4**	**69.6**	**46.7**	**56.2**	**48.2**	**56.4**

Table 5: Results comparing F1 score and exact match (EM) performance of Time-BigBird and Time-FiD for QA task on easy and hard sections of TimeQA dataset. We evaluate the Transformer models in 3 settings - fine-tune on TimeQA; fine-tune TriviaQA; and fine-tune on NQ then TimeQA. Green shows improvement due to our proposed Time-Transformer model, while we see degradation due to Euclidean variant of Time-Transformer (E)

losses for the proposed model (Table 3, highlighted in green). Learning DocTime by propagating losses over the entire document graph severely deteriorates model performance as the model has very limited training documents samples (182 for TDT, 400 for TDG, 80 for ContractTDG). Our proposed path prediction loss shows superior performance over pairwise link prediction as it jointly learns the relation label between a pair of nodes as well as the shortest dependency path linking them. As a result, the model can recover from structure prediction errors between nodes by learning an alternative path reconstructed through multi-hop connections.

6.2 Application of Temporal Dependency Parsing for downstream tasks

We train the DocTime model on the TDG corpus, which can be used to infer a temporal dependency graph from raw text samples. We extract events and timexes using CAEVO (Chambers et al., 2014) for all data samples in train,validate, and test. The temporal dependency graph acquired for each document is used as a prior for Time-Transformer to perform downstream tasks.

Performance of Time-Transformer on Temporal NLI: The temporal NLI task requires a model to identify the semantic relationship (entailed, not-entailed) between the context and corresponding hypothesis sentence based on temporal information from text. The temporal dependency graphs extracted using the DocTime trained on the TDG corpus are used as prior for Time-BERT for entailment classification. Table 4 shows the test accuracies of Time-BERT-large, Time-RoBERTa-large and other competitive baselines [(Iyyer et al., 2015),(Conneau et al., 2017)] reported by (Vashishtha et al., 2020). The temporal information prior proposed in Time-Transformer helps the BERT and RoBERTa models perform much better on the NLI task. The accuracy improved by 1.5-2.3 F1 points by applying our framework on the RoBERTa model across the five subsets. We observe the performance gain in the case of the Euclidean version of Time-RoBERTa to be modest as compared to its hyperbolic counterpart.

Performance of Time-Transformer on TimeQA: The TimeQA task focuses on understanding the time scope of facts in the long text followed by answering questions conditioned on the query and the document using implicit temporal information. We then apply the DocTime model output trained on the TDG corpus to the Time-Transformer framework on BigBird and FiD language models for long document question answering task. Following (Chen et al., 2021), we experiment with three variants of pre-trained settings: (1) fine-tuned on the TimeQA training set; (2) fine-tuned on NQ/TriviaQA data (3) fine-tuned on NQ/TriviaQA data and TimeQA.

Table 5 shows the effectiveness of Time-BigBird and Time-FiD in consistently outperforming their corresponding baselines in all three settings. More specifically, we see a realtive gain of 10-14% in F1 and exact match scores (EM) for both easy and hard sections of the dataset. It is impressive to note that the improvements due to the Time-BigBird and Time-FiD models are steady with different pre-training setups with the addition of only a few extra parameters to the baseline model. An important observation here is that the Euclidean versions of Time-BigBird and Time-FiD show persistent performance deterioration across all settings for TimeQA. We attribute this phenomenon to our initial hypothesis behind using hyperbolic operations in the proposed Temporally-informed self attention (TISA) layer. As the text length grows, the complexity of geometric operations increases, leading to vectorial distortions in Euclidean spaces (Ganea et al., 2018). This is remedied by hyperbolic transformations of masked self-attention learning in the proposed Time-Transformer.

Our experiments provide evidence that temporal dependency graphs extracted using DocTime and then utilized as a prior by temporally-

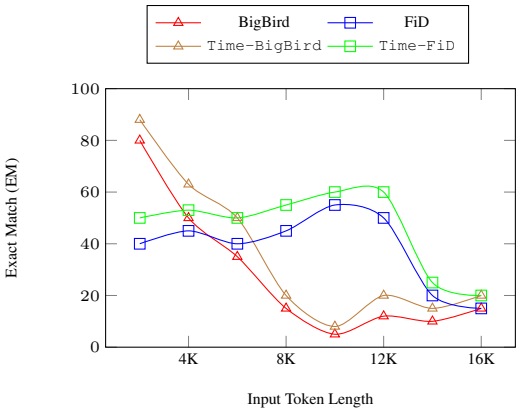

Figure 4: Impact analysis of long-distance dependencies on Transformer models for TimeQA task. Plot shows the exact match (EM) accuracy vs length of input document for hard samples. We use BigBird and FiD fine-tuned on NQ + TimeQA as backbone models. `Time-BigBird` and `Time-FiD` maintain steady improvement over baseline models even with increase in input lengths.

Corpus	Model	Structure + Relation (F1)			
		te,te	e,te	e,e	full
TD-Graphs	Heuristic	0.82	0.58	0.34	0.51
	Neural Ranking Parser (Zhang and Xue, 2018a)	0.93	0.66	0.58	0.66
	BERT Ranking Parser (Ross et al., 2020b)	0.93	0.74	0.58	0.71
	DocTime	**0.96**	**0.75**	**0.72**	**0.77**
Contract-TDG	Heuristic	0.45	0.36	0.18	0.33
	Neural Ranking Parser (Zhang and Xue, 2018a)	0.57	0.45	0.29	0.48
	BERT Ranking Parser (Ross et al., 2020b)	0.70	0.54	0.33	0.61
	DocTime	**0.75**	**0.56**	**0.39**	**0.64**

Table 6: Performance (F1 score) of `DocTime` across timex-timex, event-timex and event-event pairs for dependency structure+relation prediction on TDG and ContractTDG datasets. `DocTime` outperforms all baselines on every setting.

informed Transformer architectures such as `Time-Transformer` can improve the performance of several downstream tasks that require temporal reasoning at the sentence-level as well as at the document-level.

Impact of Long-term Dependency on `Time-Transformer` performance: We plot Fig. 4 to understand the capability of Transformer models to handle the long-term dependency in temporal reasoning on the TimeQA dataset. Plot shows the exact match (EM) accuracy vs length of the input document for hard samples. We use BigBird and FiD models fine-tuned on NQ + TimeQA as backbone models. BigBird's performance degrades rapidly as the length increases to over 5000 tokens, while the FiD's performance is quite uniformly distributed across different document lengths due to it's strong capability to deal with long-term dependency. `Time-BigBird` and `Time-FiD` follow a similar trend and maintain steady improvements over their corresponding baseline models with increasing in input lengths.

Space complexity analysis: We choose RoBERTa-base as the base model to analyze the space complexity. Liu et al. (2019b) reported the number of trainable parameters in RoBERTa-Base to be about 123 million. `Time-RoBERTa` introduces an additional 2 million parameters in total due to k-hop mask learning in the TISA layer. Therefore, Time-BERT adds few parameters to the base model without affecting its original space complexity.

Time Complexity analysis: We assume the number of tokens in each sentence to be n and extract k-hop mask matrices from a text document is $O(n^2)$ in the online inference phase. The time complexity of the Transformer embedding lookup layer is $O(n)$. The TISA layer calculates the attention score in $O(KD_q n^2)$ for both QK^T and learns the mask weights using a hyperbolic feedforward layer (MW^M), where D_q is dimension of Q and K is the number of sub-networks. The time complexity of the `Time-BERT` remains the same for small enough value of k ($k \leq 15$ in experiments).

7 Conclusion

We present `DocTime`, a new temporal dependency parsing approach that improves upon previous approaches by integrating longer term temporal information through a graph network with a novel path prediction loss. Additionally, we are able to show how a TDG can be incorporated into Transformer networks with Time-Transformer to improve on down stream tasks for NLI and question answering. Finally we introduce a TDG dataset in a new domain (Contractual documents) to expand research in this temporal reasoning to a new application domain. Future works will aim to explore more ways for integrating temporal dependency graphs into neural architectures across different application domains. In future, we would like to explore temporal event mining to aid various social media applications such as improving hate speech detection (Mathur et al., 2018b; Chopra et al., 2020), analyzing temporality in suicidal ideation detection (Mishra et al., 2019; Mathur et al., 2020) and abuse detection (Gautam et al., 2020; Sawhney et al., 2021). The proposed Time-Transformer can find applications in augmenting financial tasks (Sawhney et al., 2020), affective computing (Mittal et al., 2021), and AI for social good (Mathur et al., 2018a) with temporal common sense reasoning.

References

Mohammed Aldawsari, Adrian Perez, Deya Banisakher, and Mark Finlayson. 2020. Distinguishing between foreground and background events in news. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5171–5180.

James F Allen. 1983. Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26(11):832–843.

Jiangang Bai, Yujing Wang, Yiren Chen, Yaming Yang, Jing Bai, Jing Yu, and Yunhai Tong. 2021a. Syntax-bert: Improving pre-trained transformers with syntax trees. *arXiv preprint arXiv:2103.04350*.

Song Bai, Feihu Zhang, and Philip HS Torr. 2021b. Hypergraph convolution and hypergraph attention. *Pattern Recognition*, 110:107637.

Miguel Ballesteros, Rishita Anubhai, Shuai Wang, Nima Pourdamghani, Yogarshi Vyas, Jie Ma, Parminder Bhatia, K. McKeown, and Yaser Al-Onaizan. 2020. Severing the edge between before and after: Neural architectures for temporal ordering of events. *ArXiv*, abs/2004.04295.

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.

Parminder Bhatia, Yangfeng Ji, and Jacob Eisenstein. 2015. Better document-level sentiment analysis from rst discourse parsing. *arXiv preprint arXiv:1509.01599*.

Taylor Cassidy, Bill McDowell, Nathanel Chambers, and Steven Bethard. 2014. An annotation framework for dense event ordering. Technical report, Carnegie-Mellon Univ Pittsburgh PA.

Nathanael Chambers, Taylor Cassidy, Bill McDowell, and Steven Bethard. 2014. Dense event ordering with a multi-pass architecture. *Transactions of the Association for Computational Linguistics*, 2:273–284.

Wenhu Chen, Xinyi Wang, and William Yang Wang. 2021. A dataset for answering time-sensitive questions. *ArXiv*, abs/2108.06314.

Yu Chen, Lingfei Wu, and Mohammed Zaki. 2020. Iterative deep graph learning for graph neural networks: Better and robust node embeddings. *Advances in Neural Information Processing Systems*, 33.

Shivang Chopra, Ramit Sawhney, Puneet Mathur, and Rajiv Ratn Shah. 2020. Hindi-english hate speech detection: Author profiling, debiasing, and practical perspectives. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 386–393.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loic Barrault, and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data. *arXiv preprint arXiv:1705.02364*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. *ArXiv*, abs/1810.04805.

Yifan Feng, Haoxuan You, Zizhao Zhang, Rongrong Ji, and Yue Gao. 2019. Hypergraph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3558–3565.

Octavian-Eugen Ganea, Gary Bécigneul, and Thomas Hofmann. 2018. Hyperbolic neural networks. *arXiv preprint arXiv:1805.09112*.

Hongyang Gao and Shuiwang Ji. 2019. Graph u-nets. In *international conference on machine learning*, pages 2083–2092. PMLR.

Akash Gautam, Puneet Mathur, Rakesh Gosangi, Debanjan Mahata, Ramit Sawhney, and Rajiv Ratn Shah. 2020. # metooma: Multi-aspect annotations of tweets related to the metoo movement. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 14, pages 209–216.

Tanya Goyal and Greg Durrett. 2019. Embedding time expressions for deep temporal ordering models. In *ACL*.

Rujun Han, I-Hung Hsu, Mu Yang, A. Galstyan, R. Weischedel, and Nanyun Peng. 2019a. Deep structured neural network for event temporal relation extraction. In *CoNLL*.

Rujun Han, Mengyue Liang, Bashar Alhafni, and Nanyun Peng. 2019b. Contextualized word embeddings enhanced event temporal relation extraction for story understanding. *ArXiv*, abs/1904.11942.

Rujun Han, Qiang Ning, and Nanyun Peng. 2019c. Joint event and temporal relation extraction with shared representations and structured prediction. In *EMNLP/IJCNLP*.

Rujun Han, Yichao Zhou, and Nanyun Peng. 2020. Domain knowledge empowered structured neural net for end-to-end event temporal relation extraction. *ArXiv*, abs/2009.07373.

Dan Hendrycks, Collin Burns, Anya Chen, and Spencer Ball. 2021. Cuad: An expert-annotated nlp dataset for legal contract review. *arXiv preprint arXiv:2103.06268*.

John Hewitt and Christopher D Manning. 2019. A structural probe for finding syntax in word representations. 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)*, pages 4129–4138.

Mohit Iyyer, Varun Manjunatha, Jordan Boyd-Graber, and Hal Daumé III. 2015. Deep unordered composition rivals syntactic methods for text classification. In *Proceedings of the 53rd annual meeting of the association for computational linguistics and the 7th international joint conference on natural language processing (volume 1: Long papers)*, pages 1681–1691.

Gautier Izacard and Edouard Grave. 2020. Leveraging passage retrieval with generative models for open domain question answering. *arXiv preprint arXiv:2007.01282*.

Gautier Izacard and Edouard Grave. 2021. Leveraging passage retrieval with generative models for open domain question answering. In *EACL*.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does bert learn about the structure of language? In *ACL 2019-57th Annual Meeting of the Association for Computational Linguistics*.

Woojeong Jin, Rahul Khanna, Suji Kim, Dong-Ho Lee, Fred Morstatter, Aram Galstyan, and Xiang Ren. 2020. Forecastqa: A question answering challenge for event forecasting with temporal text data. *arXiv preprint arXiv:2005.00792*.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S Weld, Luke Zettlemoyer, and Omer Levy. 2020. Spanbert: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77.

Eliyahu Kiperwasser and Miguel Ballesteros. 2018. Scheduled multi-task learning: From syntax to translation. *Transactions of the Association for Computational Linguistics*, 6:225–240.

Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Oleksandr Kolomiyets, Steven Bethard, and Marie Francine Moens. 2012. 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.

A. Leeuwenberg and Marie-Francine Moens. 2018. Temporal information extraction by predicting relative time-lines. *ArXiv*, abs/1808.09401.

A. Leeuwenberg and Marie-Francine Moens. 2019. A survey on temporal reasoning for temporal information extraction from text. *ArXiv*, abs/2005.06527.

Yuxiao Lin, Yuxian Meng, Xiaofei Sun, Qinghong Han, Kun Kuang, Jiwei Li, and Fei Wu. 2021. Bertgcn: Transductive text classification by combining gcn and bert. *arXiv preprint arXiv:2105.05727*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019a. Roberta: A robustly optimized bert pretraining approach. *ArXiv*, abs/1907.11692.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Zhengyuan Liu, Ke Shi, and Nancy F Chen. 2021. Coreference-aware dialogue summarization. *arXiv preprint arXiv:2106.08556*.

Puneet Mathur, Meghna Ayyar, Sahil Chopra, Simra Shahid, Laiba Mehnaz, and Rajiv Shah. 2018a. Identification of emergency blood donation request on twitter. In *Proceedings of the 2018 EMNLP Workshop SMM4H: The 3rd Social Media Mining for Health Applications Workshop & Shared Task*, pages 27–31.

Puneet Mathur, Rajiv Jain, Franck Dernoncourt, Vlad Morariu, Quan Hung Tran, and Dinesh Manocha. 2021a. 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)*, pages 524–533, Online. Association for Computational Linguistics.

Puneet Mathur, Rajiv Jain, Franck Dernoncourt, Vlad Morariu, Quan Hung Tran, and Dinesh Manocha. 2021b. 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)*, pages 524–533.

Puneet Mathur, Ramit Sawhney, Shivang Chopra, Maitree Leekha, and Rajiv Ratn Shah. 2020. Utilizing temporal psycholinguistic cues for suicidal intent estimation. In *European Conference on Information Retrieval*, pages 265–271. Springer.

Puneet Mathur, Rajiv Shah, Ramit Sawhney, and Debanjan Mahata. 2018b. Detecting offensive tweets in hindi-english code-switched language. In *Proceedings of the Sixth International Workshop on Natural Language Processing for Social Media*, pages 18–26.

Rohan Mishra, Pradyumn Prakhar Sinha, Ramit Sawhney, Debanjan Mahata, Puneet Mathur, and Rajiv Ratn Shah. 2019. Snap-batnet: Cascading author profiling and social network graphs for suicide ideation detection on social media. In *Proceedings of the 2019 conference of the North American Chapter of the association for computational linguistics: student research workshop*, pages 147–156.

Trisha Mittal, Puneet Mathur, Aniket Bera, and Dinesh Manocha. 2021. Affect2mm: Affective analysis of multimedia content using emotion causality. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5661–5671.

Qiang Ning, Sanjay Subramanian, and D. Roth. 2019a. An improved neural baseline for temporal relation extraction. In *EMNLP/IJCNLP*.

Qiang Ning, Sanjay Subramanian, and Dan Roth. 2019b. An improved neural baseline for temporal relation extraction. *arXiv preprint arXiv:1909.00429*.

Qiang Ning, Hao Wu, Rujun Han, Nanyun Peng, Matt Gardner, and Dan Roth. 2020a. Torque: A reading comprehension dataset of temporal ordering questions. In *EMNLP*.

Qiang Ning, Hao Wu, Rujun Han, Nanyun Peng, Matt Gardner, and Dan Roth. 2020b. Torque: A reading comprehension dataset of temporal ordering questions. *arXiv preprint arXiv:2005.00242*.

Qiang Ning, Hao Wu, and Dan Roth. 2018. A multi-axis annotation scheme for event temporal relations. In *ACL*.

Yunseok Noh, Yong-Min Shin, Junmo Park, A.-Yeong Kim, Su Jeong Choi, Hyun-Je Song, Seongbae Park, and Seyoung Park. 2020. Wire: An automated report generation system using topical and temporal summarization. *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*.

James Pustejovsky, Patrick Hanks, Roser Sauri, Andrew See, Robert Gaizauskas, Andrea Setzer, Dragomir Radev, Beth Sundheim, David Day, Lisa Ferro, et al. 2003a. The timebank corpus. In *Corpus linguistics*, volume 2003, page 40. Lancaster, UK.

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

James Pustejovsky and Amber Stubbs. 2011. Increasing informativeness in temporal annotation. In *Proceedings of the 5th Linguistic Annotation Workshop*, pages 152–160.

Lianhui Qin, Aditya Gupta, Shyam Upadhyay, Luheng He, Yejin Choi, and Manaal Faruqui. 2021. Timedial: Temporal commonsense reasoning in dialog. *arXiv preprint arXiv:2106.04571*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. 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 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Hayley Ross, Jonathon Cai, and Bonan Min. 2020a. 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, Online. Association for Computational Linguistics.

Hayley Ross, Jonathon Cai, and Bonan Min. 2020b. Exploring contextualized neural language models for temporal dependency parsing. *arXiv preprint arXiv:2004.14577*.

Ramit Sawhney, Puneet Mathur, Taru Jain, Akash Kumar Gautam, and Rajiv Shah. 2021. Multitask learning for emotionally analyzing sexual abuse disclosures. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4881–4892.

Ramit Sawhney, Puneet Mathur, Ayush Mangal, Piyush Khanna, Rajiv Ratn Shah, and Roger Zimmermann. 2020. Multimodal multi-task financial risk forecasting. In *Proceedings of the 28th ACM international conference on multimedia*, pages 456–465.

Ke Shi, Zhengyuan Liu, and Nancy F Chen. 2020. An end-to-end document-level neural discourse parser exploiting multi-granularity representations. *arXiv preprint arXiv:2012.11169*.

Pontus Stenetorp, Sampo Pyysalo, Goran Topić, Tomoko Ohta, Sophia Ananiadou, and Jun'ichi Tsujii. 2012. Brat: a web-based tool for nlp-assisted text annotation. In *Proceedings of the Demonstrations at the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 102–107.

Jannik Strötgen and Michael Gertz. 2013. Multilingual and cross-domain temporal tagging. *Language Resources and Evaluation*, 47(2):269–298.

Siddharth Vashishtha, Adam Poliak, Yash Kumar Lal, Benjamin Van Durme, and Aaron Steven White. 2020. Temporal reasoning in natural language inference. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4070–4078, Online. Association for Computational Linguistics.

Jiarui Yao, Haoling Qiu, Bonan Min, and Nianwen Xue. 2020a. Annotating temporal dependency graphs via crowdsourcing. In *EMNLP*.

Jiarui Yao, Haoling Qiu, Bonan Min, and Nianwen Xue. 2020b. Annotating Temporal Dependency Graphs via Crowdsourcing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5368–5380, Online. Association for Computational Linguistics.

Liang Yao, Chengsheng Mao, and Yuan Luo. 2019. Graph convolutional networks for text classification. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 7370–7377.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontañón, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Amr Ahmed. 2020a. Big bird: Transformers for longer sequences. *ArXiv*, abs/2007.14062.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. 2020b. Big bird: Transformers for longer sequences. In *NeurIPS*.

Yuchen Zhang and Nianwen Xue. 2018a. Neural ranking models for temporal dependency structure parsing. *arXiv preprint arXiv:1809.00370*.

Yuchen Zhang and Nianwen Xue. 2018b. Structured interpretation of temporal relations. *arXiv preprint arXiv:1808.07599*.

Yuchen Zhang and Nianwen Xue. 2019. Acquiring structured temporal representation via crowdsourcing: A feasibility study. In **SEMEVAL*.

Zhenyu Zhang, Bowen Yu, Xiaobo Shu, Tingwen Liu, Hengzhu Tang, Wang Yubin, and Li Guo. 2020. Document-level relation extraction with dual-tier heterogeneous graph. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1630–1641.

Xinyu Zhao, Shih-ting Lin, and Greg Durrett. 2020. Effective distant supervision for temporal relation extraction. *arXiv preprint arXiv:2010.12755*.

A Ethics Statement

We utilize two publicly available datasets - TDT and TDG for evaluating temporal dependency parser. We also curated dataset for TDG on contract documents. We source these contract documents from a publicly available resource - ATTICUS. We repurpose the document in this dataset for our task and provide new annotations. ContractTDG dataset does not violate any privacy as these documents are already in public domain. There is no human bias involved in such documents as they are business contracts filed on the SEC website. These documents do not restrict reuse for academic purposes and any personal information was already redacted before their original release. All documents and our experiments are restricted to English language. Temporal NLI and TimeQA datasets that are publicly available for research purposes. The crowd workers are paid a fair wage. There was no sensitive data involved in the studies.

B Details on Graph Feature Extraction

B.1 Structural Graph Features

The Structural Graph (G_{str}) enriches the token level features with a hierarchical textual structure formed by grouping word tokens into lists of sentences that bind together to form the text document. Prior work has shown that transductive graph learning over G_{str} can help learn the long range word-word dependencies set several sentences apart through hierarchical text modeling (Yao et al., 2019). The directed edges of the Structural Graph encode the following relationships: **(1) Document-Sentence Affiliation**, which connects each document-node to a sentence-node; **(2) Sentence-Word Affiliation**, which joins each sentence node to its constituent word nodes; **(3) Sentence-Sentence Adjacency** and **(4) Word-Word Adjacency**, which preserve sequential ordering for consecutive sentence and word nodes, respectively. For the structural graph, a sentence node embedding s_i is obtained by passing sentences through a pre-trained SentenceBERT model (Reimers and Gurevych, 2019) and the document node embedding D is calculated as the average of all sentence embeddings ($D = \sum_{i=0}^{m} v_i$).

BertGCN (Lin et al., 2021) combines the advantages of both large-scale pre-training and transductive learning. We input the structural graph G_{str} to BertGCN model[7] where each node represents a word, a sentence or the document. BertGCN processes the input node feature matrix sequentially through a Bert model to fine-tune each node to learn local contextual representations. This is followed by passing the learned node feature matrix through two layers of graph convolution to take advantage of global influence propagation through graph edges across multi-hop nodes.

B.2 Syntactic Graph Features

Syntactic cues are useful priors for learning based NLP tasks (Kiperwasser and Ballesteros, 2018). Pre-trained transformer models can capture certain syntactic information implicitly (Hewitt and Manning, 2019) but Jawahar et al. (2019) showed that BERT needs to be trained with deeper layers for handling harder cases involving long-distance dependency information. Moreover, past studies have pointed to the existence of multi-hop coreferring expressions in document-level text due to anaphora and cataphora (Joshi et al., 2020).

G_{syn} is made of separate nodes to represent each constituent word w_i in the document. For each document, there is also a set of co-reference clusters $\{\mathtt{C}_1, \mathtt{C}_2, \cdots, \mathtt{C}_u\}$ referring to the same entities in the graph. We define four types of directed edges in G_{syn} as described below where ξ denotes the set of syntactic dependency arcs inside sentences, $S^r_{w_i}$ denotes root of the sentence in which w_i belongs, and $S_{w_i} \rightarrow S_{w_j}$ represents whether sentences containing words w_i and w_j are adjacent.

$$\varepsilon_{syn}(i,j) = \begin{cases} \text{dependency} & \text{if } (w_i, w_j) \in \xi \\ \text{reversion} & \text{if } (w_j, w_i) \in \xi \\ \text{coreference} & \text{if } w_i, w_j \in \mathtt{C}_u \\ \text{self-loop} & \text{if } i == j \\ \text{root-adjacency} & \text{if } w_i == S^r_{w_i}, \\ & \& \; w_j == S^r_{w_j}, \\ & \& \; S_{w_i} \rightarrow S_{w_j} \end{cases} \tag{3}$$

The first two edge types are introduced to allow information flow along and against syntactic arcs between intra-sentential dependency relations to enrich contextually learned embeddings of each word. We connect parse tree roots of adjacent sentences to encode document level long-range syntactic relatedness between sentences. We add an undirected edge between word nodes if both belong to the same co-reference cluster. Inspired by (Kipf and Welling, 2016), self-loop edges are added for better message passing iterations. G_{syn} is instantiated

[7] Implementation Used: https://github.com/ZeroRin/BertGCN

as a gated variant of Weighted Relational Graph Convolutional Network (WR-GCN) (Zhang et al., 2020) with k-layers. WR-GCN can able to model diverse relations in a heterogeneous graph by treating different types of edges with unequal weights assigned during message passing.

B.3 Semantic Graph Features

Semantic Role Labeling (SRL) parses text sequences to recognize the predicate-argument structure in the sentence to answer *who* did *what* and *when*. Anchoring verb events to their temporal argument spans extracted from semantic parsing helps infer event relationships with their associated time expressions. This can be complemented by discourse features in the form of RST connections can help leverage long-range document-level interactions between phrase units (Bhatia et al., 2015) and identify background-foreground events(Aldawsari et al., 2020) and improve temporal relationship parsing (Mathur et al., 2021b). We utilize Document-level Rhetorical Structure Theory (RST) parser (Shi et al., 2020) to organize contiguous semantic text spans of a document into a hierarchical dependency structure labeled with their rhetorical relations.

G_{sem} consists of individual nodes for each constituent word w_i in the document. Discourse units and temporal arguments may span several word tokens $\{w_1, w_2, \cdots w_k\}$. We add two types of directed edge connections between - (1) event verb predicate - temporal argument edge (ε_t) such that $(w_e \rightarrow \{w_1, w_2, \cdots w_k\} \in \varepsilon_t)$; (2) Rhetorical pair edges (ε_d) labelled by the type of the rhetorical relation $(\{w_1, w_2, \cdots w_i\} \rightarrow \{w_1, w_2, \cdots w_j\} \in \varepsilon_d)$.

$$\varepsilon = \begin{cases} w_e \rightarrow \{w_e, \cdots w_k\} \in \varepsilon_t \\ \{w_1, \cdots w_i\} \rightarrow \{w_1, \cdots w_j\} \in \varepsilon_d \end{cases} \quad (4)$$

The nature of edge connections in G_{sem} extends beyond pairwise interactions as each edge may connect to one or more word nodes. Hence, we formulate the semantic graph as a hypergraph (Feng et al., 2019) where an edge can join an arbitrary number of vertices. We construct $G_{sem} = (\nu, \varepsilon, \mathbf{W})$ where ν is the set of all word nodes w_i, and ε is the subset of hyperedges such that $\varepsilon = \varepsilon_t \cup \varepsilon_d$. Each hyperedge e is assigned a positive weight corresponding to the type of edge relation and is stored in a diagonal matrix $\mathbf{W} \in \Re^{|\varepsilon| x |\varepsilon|}$. The semantic graph is learned using hypergraph convolution layers (Bai et al., 2021b) to obtain discriminative node embeddings for each word node.

C Training Setup

Hyperparameter: Hyper-parameters for `DocTime` were tuned on the respective validation set to find the best configurations for different datasets. We summarize the range of our model's hyper parameters such as: number of hidden layers in WR-GCN/BERT-GCN/HyperGraphGCN $\{1, 2, 3\}$, size of hidden layers in WR-GCN/BERT-GCN/HyperGraphGCN $\{64, 128, 256, 512\}$, BERT embedding size (768), dropout $\delta \in \{0.2, 0.3, 0.4, 0.5.0.6\}$, learning rate $\lambda \in \{1e-5, 1e-4, 1e-3, 1e-2, 1e-1\}$, weight decay $\omega \in \{1e-6, 1e-5, 1e-4, 1e-3\}$, batch size $b \in \{16, 32, 64\}$ and epochs ($\leq$ 100), ϵ-sparsity $\in [0, 1]$, IDGL smoothness ratio=0.5, IDGL sparsity ratio=0.5, IDGL connectivity ratio=0.5, size of hidden layers in Graph U-net $\{64, 128, 256, 512\}$.

Loss Function and Inference: `Time-Transformers` are trained using Cross Entropy loss with Adam optimizer. Across both TempNLI and TimeQA datasets, we found the best results correspond with the use of Adam optimiser set with default values $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 1e-8$, weight-decay of $5e-4$ and an initial learning rate of 0.001.

`DocTime` uses cross entropy loss for structure prediction. For structure+relation classification, it uses the path prediction loss as defined in Methodology.

Computing Infrastructue: `DocTime` and `Time-Tranformers` are written in PyTorch library and were trained on 4 and 6 Nvidia GeForce RTX 2080 GPU, respectively. **Average Runtime**: `DocTime` takes a maximum of approximately 5 hrs to train once on TDG datasets. `Time-BERT`, `Time-RoBERTa` take 3 hrs to fine-tune on TempNLI. `Time-BigBird`, `Time-FiD` takes 8,12 hours to fine-tune, respectively.

Dataset Access

Links to download TDT dataset: https://github.com/yuchenz/crowdsourced_EN_TDT_corpus

Link to download TDG dataset: https://github.com/Jryao/temporal_dependency_graphs_crowdsourcing

Link to download Temporal NLI dataset: https://github.com/sidsvash26/temporal_nli

Link to download TimeQA dataset: https://

Corpus	Model	Structure + Relation (F1)			
		te,te	e,te	e,e	full
TD-Graphs	Heuristic	0.82	0.58	0.34	0.51
	Neural Ranking Parser (Zhang and Xue, 2018a)	0.93	0.66	0.58	0.66
	BERT Ranking Parser (Ross et al., 2020b)	0.93	0.74	0.58	0.71
	DocTime	**0.96**	**0.75**	**0.72**	**0.77**
Contract-TDG	Heuristic	0.45	0.36	0.18	0.33
	Neural Ranking Parser (Zhang and Xue, 2018a)	0.57	0.45	0.29	0.48
	BERT Ranking Parser (Ross et al., 2020b)	0.70	0.54	0.33	0.61
	DocTime	**0.75**	**0.56**	**0.39**	**0.64**

Table 7: Performance (F1 score) of DocTime across timex-timex, event-timex and event-event pairs for dependency structure+relation prediction on TDG and ContractTDG datasets. DocTime outperforms all baselines on every setting.

github.com/wenhuchen/Time-Sensitive-QA

D Hyperparameters

Table 8 show the Training hyperparameters of DocTime for TDT, TDG, ContractTDG datasets.

E More Results

Performance across different relation types: We analyze the benefits of DocTime for different types of relations in document-level TDG datasets in Table 7. We report F1 scores for structure+relation prediction for timex-timex, event-timex and event-event pairs. We observe a relatively smaller performance gap between the BERT Ranking parser and DocTime for event-timex relations. However, DocTime shows relatively stronger performance for event-event relations. This phenomenon can be attributed to the fact that both datasets tend to have event-event links between event pairs that are on an average closer in word distance, whereas a higher ratio of event-timex and timex-timex pairs are several sentences apart. DocTime can integrate long-range inter-dependencies between entity pairs that are several sentences (or paragraphs in Contract TDG) apart.

	Dataset		
Hyperparameters	TDT	TDG	Contract
Dropout Ratio	0.5	0.5	0.5
Optimizer	Adam	Adam	Adam
Input Dimension (Structural Graph)	(n,768)	(n,768)	(n,768)
Input Dimension (Syntactic Graph)	(n,768)	(n,768)	(n,768)
Input Dimension (Semantic Graph)	(n,768)	(n,768)	(n,768)
Hidden Dimension (WR-GCN)	256	256	64
Number of hidden layers (WR-GCN)	2	2	2
Hidden Dimension (BERT-GCN)	256	256	64
Number of hidden layers (BERT-GCN)	1	1	1
Hidden Dimension (HyperGCN)	256	256	64
Number of hidden layers (HyperGCN)	2	2	2
Epochs	20	20	20
Batch Size	8	8	16
Learning Rate	2e-5	2e-5	2e-5
Activation Function of Linear layers	ReLU	ReLU	ReLU

Table 8: **Hyperparameters Details:** Training hyperparameters of `DocTime` for TDT, TDG, ContractTDG