Dating Documents using Graph Convolution Networks

Shikhar Vashishth
IISc Bangalore
shikhar@iisc.ac.in

Shib Sankar Dasgupta
IISc Bangalore
shibd@iisc.ac.in

Swayambhu Nath Ray
IISc Bangalore
swayambhuray@iisc.ac.in

Partha Talukdar
IISc Bangalore
ppt@iisc.ac.in

Abstract

Document date is essential for many important tasks, such as document retrieval, summarization, event detection, etc. While existing approaches for these tasks assume accurate knowledge of the document date, this is not always available, especially for arbitrary documents from the Web. Document Dating is a challenging problem which requires inference over the temporal structure of the document. Prior document dating systems have largely relied on handcrafted features while ignoring such document-internal structures. In this paper, we propose NeuralDater, a Graph Convolutional Network (GCN) based document dating approach which jointly exploits syntactic and temporal graph structures of document in a principled way. To the best of our knowledge, this is the first application of deep learning for the problem of document dating. Through extensive experiments on real-world datasets, we find that NeuralDater significantly outperforms state-of-the-art baseline by 19% absolute (45% relative) accuracy points.

1 Introduction

Date of a document, also referred to as the Document Creation Time (DCT), is at the core of many important tasks, such as, information retrieval (Olson et al., 1999; Li and Croft, 2003; Dakka et al., 2008), temporal reasoning (Mani and Wilson, 2000; Llidó et al., 2001), text summarization (Wan, 2007), event detection (Allan et al., 1998), and analysis of historical text (de Jong et al., 2005a), among others. In all such tasks, the document date is assumed to be available and also accurate – a strong assumption, especially for arbitrary documents from the Web. Thus, there is a need to automatically predict the date of a document based on its content. This problem is referred to as *Document Dating*.

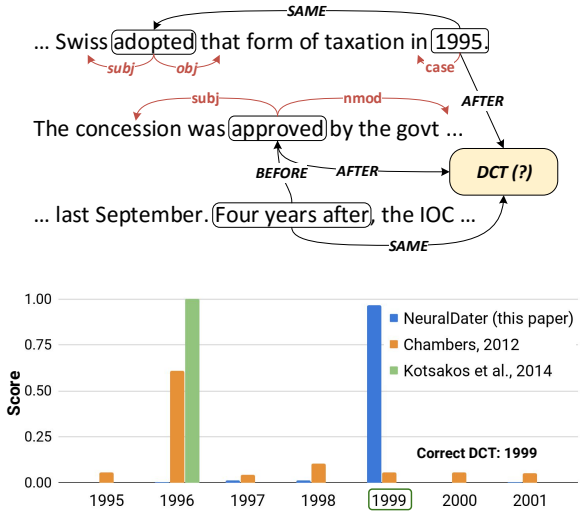

Figure 1: **Top:** An example document annotated with syntactic and temporal dependencies. In order to predict the right value of 1999 for the Document Creation Time (DCT), inference over these document structures is necessary. **Bottom:** Document date prediction by two state-of-the-art-baselines and NeuralDater, the method proposed in this paper. While the two previous methods are getting misled by the temporal expression (*1995*) in the document, NeuralDater is able to use the syntactic and temporal structure of the document to predict the right value (*1999*).

Initial attempts on automatic document dating started with generative models by (de Jong et al., 2005b). This model is later improved by (Kanhabua and Nørvåg, 2008a) who incorporate additional features such as POS tags, collocations, etc. Chambers (2012) shows significant improvement over these prior efforts through their discriminative models using handcrafted temporal features. Kotsakos et al. (2014) propose a statistical approach for document dating exploiting term bursti-

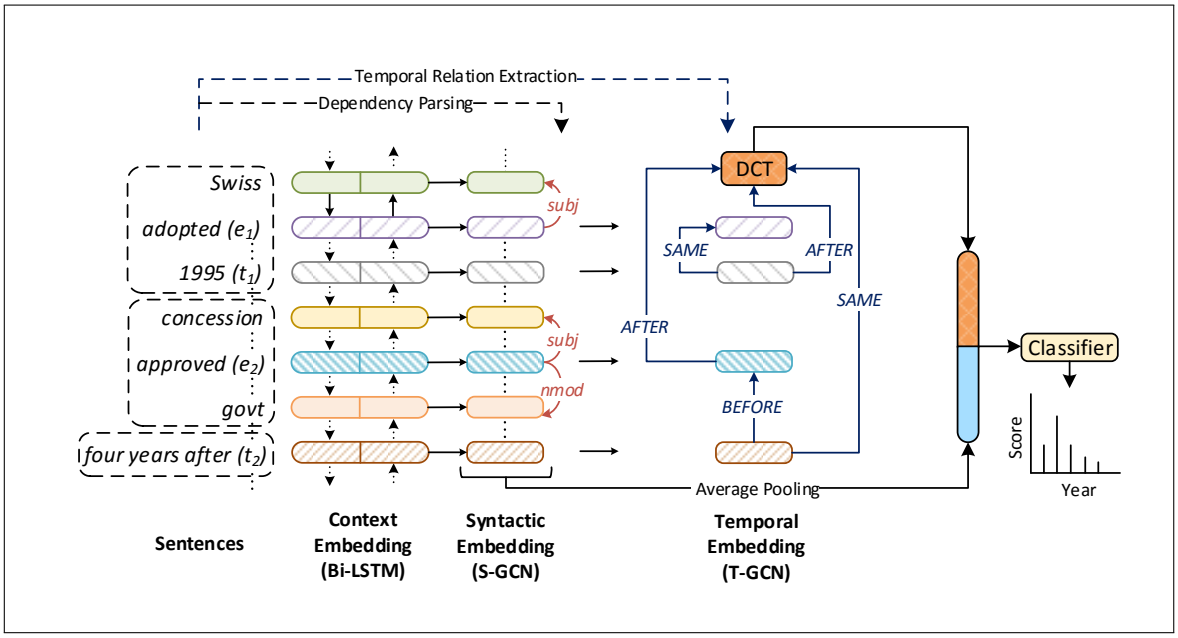

Figure 2: Overview of NeuralDater. NeuralDater exploits syntactic and temporal structure in a document to learn effective representation, which in turn are used to predict the document time. NeuralDater uses a Bi-directional LSTM (Bi-LSTM), two Graph Convolution Networks (GCN) – one over the dependency tree and the other over the document's temporal graph – along with a softmax classifier, all trained end-to-end jointly. Please see Section 4 for more details.

ness (Lappas et al., 2009).

Document dating is a challenging problem which requires extensive reasoning over the temporal structure of the document. Let us motivate this through an example shown in Figure 1. In the document, *four years after* plays a crucial role in identifying the creation time of the document. The existing approaches give higher confidence for timestamp immediate to the year mention *1995*. NeuralDater exploits the syntactic and temporal structure of the document to predict the right timestamp (1999) for the document. With the exception of (Chambers, 2012), all prior works on the document dating problem ignore such informative temporal structure within the document.

Research in document event extraction and ordering have made it possible to extract such temporal structures involving events, temporal expressions, and the (unknown) document date in a document (Mirza and Tonelli, 2016; Chambers et al., 2014). While methods to perform reasoning over such structures exist (Verhagen et al., 2007, 2010; UzZaman et al., 2013; Llorens et al., 2015; Pustejovsky et al., 2003), none of them have exploited advances in deep learning (Krizhevsky et al., 2012; Hinton et al., 2012; Goodfellow et al., 2016). In particular, recently proposed Graph Convolution Networks (GCN) (Defferrard et al., 2016; Kipf and Welling, 2017) have emerged as a way to learn graph representation while encoding structural information and constraints represented by the graph. We adapt GCNs for the document dating problem and make the following contributions:

- We propose NeuralDater, a Graph Convolution Network (GCN)-based approach for document dating. To the best of our knowledge, this is the first application of GCNs, and more broadly deep neural network-based methods, for the document dating problem.
- NeuralDater is the first document dating approach which exploits syntactic as well temporal structure of the document, all within a principled joint model.
- Through extensive experiments on multiple real-world datasets, we demonstrate NeuralDater's effectiveness over state-of-the-art baselines.

NeuralDater's source code and datasets used in the paper are available at `http://github.com/malllabiisc/NeuralDater`.

2 Related Work

Automatic Document Dating: de Jong et al. (2005b) propose the first approach for automating document dating through a statistical language

model. Kanhabua and Nørvåg (2008a) further extend this work by incorporating semantic-based preprocessing and temporal entropy (Kanhabua and Nørvåg, 2008b) based term-weighting. Chambers (2012) proposes a MaxEnt based discriminative model trained on hand-crafted temporal features. He also proposes a model to learn probabilistic constraints between year mentions and the actual creation time of the document. We draw inspiration from his work for exploiting temporal reasoning for document dating. Kotsakos et al. (2014) propose a purely statistical method which considers lexical similarity alongside burstiness (Lappas et al., 2009) of terms for dating documents. To the best of our knowledge, NeuralDater, our proposed method, is the first method to utilize deep learning techniques for the document dating problem.

Event Ordering Systems: Temporal ordering of events is a vast research topic in NLP. The problem is posed as a temporal relation classification between two given temporal entities. Machine Learned classifiers and well crafted linguistic features for this task are used in (Chambers et al., 2007; Mirza and Tonelli, 2014). D'Souza and Ng (2013) use a hybrid approach by adding 437 hand-crafted rules. Chambers and Jurafsky (2008); Yoshikawa et al. (2009) try to classify with many more temporal constraints, while utilizing integer linear programming and Markov logic.

CAEVO, a CAscading EVent Ordering architecture (Chambers et al., 2014) use sieve-based architecture (Lee et al., 2013) for temporal event ordering for the first time. They mix multiple learners according to their precision based ranks and use transitive closure for maintaining consistency of temporal graph. Mirza and Tonelli (2016) recently propose CATENA (CAusal and TEmporal relation extraction from NAtural language texts), the first integrated system for the temporal and causal relations extraction between pre-annotated events and time expressions. They also incorporate sieve-based architecture which outperforms existing methods in temporal relation classification domain. We make use of CATENA for temporal graph construction in our work.

Graph Convolutional Networks (GCN): GCNs generalize Convolutional Neural Network (CNN) over graphs. GCN is introduced by (Bruna et al., 2014), and later extended by (Defferrard et al., 2016) with efficient localized filter approximation in spectral domain. Kipf and Welling (2017) propose a first-order approximation of localized filters through layer-wise propagation rule. GCNs over syntactic dependency trees have been recently exploited in the field of semantic-role labeling (Marcheggiani and Titov, 2017), neural machine translation (Bastings et al., 2017a), event detection (Bastings et al., 2017b). In our work, we successfully use GCNs for document dating.

3 Background: Graph Convolution Networks (GCN)

In this section, we provide an overview of Graph Convolution Networks (GCN) (Kipf and Welling, 2017). GCN learns an embedding for each node of the graph it is applied over. We first present GCN for undirected graphs and then move onto GCN for directed graph setting.

3.1 GCN on Undirected Graph

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be an undirected graph, where $\mathcal{V}$ is a set of n vertices and $\mathcal{E}$ the set of edges. The input feature matrix $\mathcal{X} \in \mathbb{R}^{n \times m}$ whose rows are input representation of node u, $x_u \in \mathbb{R}^m, \forall u \in \mathcal{V}$. The output hidden representation $h_v \in \mathbb{R}^d$ of a node v after a single layer of graph convolution operation can be obtained by considering only the immediate neighbors of v. This can be formulated as:

$$h_v = f\left(\sum_{u \in \mathcal{N}(v)} (W x_u + b)\right), \quad \forall v \in \mathcal{V}.$$

Here, model parameters $W \in \mathbb{R}^{d \times m}$ and $b \in \mathbb{R}^d$ are learned in a task-specific setting using first-order gradient optimization. $\mathcal{N}(v)$ refers to the set of neighbors of v and f is any non-linear activation function. We have used ReLU as the activation function in this paper[1].

In order to capture nodes many hops away, multiple GCN layers may be stacked one on top of another. In particular, h_v^{k+1}, representation of node v after k^{th} GCN layer can be formulated as:

$$h_v^{k+1} = f\left(\sum_{u \in \mathcal{N}(v)} \left(W^k h_u^k + b^k\right)\right), \forall v \in \mathcal{V}.$$

where h_u^k is the input to the k^{th} layer.

[1] ReLU: $f(x) = \max(0, x)$

3.2 GCN on Labeled and Directed Graph

In this section, we consider GCN formulation over graphs where each edge is labeled as well as directed. In this setting, an edge from node u to v with label $l(u,v)$ is denoted as $(u,v,l(u,v))$. While a few recent works focus on GCN over directed graphs (Yasunaga et al., 2017; Marcheggiani and Titov, 2017), none of them consider labeled edges. We handle both direction and label by incorporating label and direction specific filters.

Based on the assumption that the information in a directed edge need not only propagate along its direction, following (Marcheggiani and Titov, 2017) we define an updated edge set $\mathcal{E}'$ which expands the original set $\mathcal{E}$ by incorporating inverse, as well self-loop edges.

$$\mathcal{E}' = \mathcal{E} \cup \{(v,u,l(u,v)^{-1}) \mid (u,v,l(u,v)) \in \mathcal{E}\} \cup \{(u,u,\top) \mid u \in \mathcal{V})\}. \quad (1)$$

Here, $l(u,v)^{-1}$ is the inverse edge label corresponding to label $l(u,v)$, and $\top$ is a special empty relation symbol for self-loop edges. We now define h_v^{k+1} as the embedding of node v after k^{th} GCN layer applied over the directed and labeled graph as:

$$h_v^{k+1} = f\left(\sum_{u \in \mathcal{N}(v)} \left(W_{l(u,v)}^k h_u^k + b_{l(u,v)}^k\right)\right). \quad (2)$$

We note that the parameters $W_{l(u,v)}^k$ and $b_{l(u,v)}^k$ in this case are edge label specific.

3.3 Incorporating Edge Importance

In many practical settings, we may not want to give equal importance to all the edges. For example, in case of automatically constructed graphs, some of the edges may be erroneous and we may want to automatically learn to discard them. Edge-wise gating may be used in a GCN to give importance to relevant edges and subdue the noisy ones. Bastings et al. (2017b); Marcheggiani and Titov (2017) used gating for similar reasons and obtained high performance gain. At k^{th} layer, we compute gating value for a particular edge $(u,v,l(u,v))$ as:

$$g_{u,v}^k = \sigma\left(h_u^k \cdot \hat{w}_{l(u,v)}^k + \hat{b}_{l(u,v)}^k\right),$$

where, $\sigma(\cdot)$ is the sigmoid function, $\hat{w}_{l(u,v)}^k$ and $\hat{b}_{l(u,v)}^k$ are label specific gating parameters. Thus, gating helps to make the model robust to the noisy labels and directions of the input graphs. GCN embedding of a node while incorporating edge gating may be computed as follows.

$$h_v^{k+1} = f\left(\sum_{u \in \mathcal{N}(v)} g_{u,v}^k \times \left(W_{l(u,v)}^k h_u^k + b_{l(u,v)}^k\right)\right).$$

4 NeuralDater Overview

The Documents Dating problem may be cast as a multi-class classification problem (Kotsakos et al., 2014; Chambers, 2012). In this section, we present an overview of NeuralDater, the document dating system proposed in this paper. Architectural overview of NeuralDater is shown in Figure 2.

NeuralDater is a deep learning-based multi-class classification system. It takes in a document as input and returns its predicted date as output by exploiting the syntactic and temporal structure of document.

NeuralDater network consists of three layers which learns an embedding for the Document Creation Time (DCT) node corresponding to the document. This embedding is then fed to a softmax classifier which produces a distribution over timestamps. Following prior research (Chambers, 2012; Kotsakos et al., 2014), we work with year granularity for the experiments in this paper. We however note that NeuralDater can be trained for finer granularity with appropriate training data. The NeuralDater network is trained end-to-end using training data. We briefly present NeuralDater's various components below. Each component is described in greater detail in subsequent sections.

- **Context Embedding**: In this layer, NeuralDater uses a Bi-directional LSTM (Bi-LSTM) to learn embedding for each token in the document. Bi-LSTMs have been shown to be quite effective in capturing local context inside token embeddings (Sutskever et al., 2014).

- **Syntactic Embedding**: In this step, NeuralDater revises token embeddings from previous step by running a GCN over the dependency parses of sentences in the document. We refer to this GCN as **Syntactic GCN** or **S-GCN**. While the Bi-LSTM captures immediate local context in token embeddings, S-

GCN augments them by capturing syntactic context.

- **Temporal Embedding**: In this step, NeuralDater further refines embeddings learned by S-GCN to incorporate cues from temporal structure of event and times in the document. NeuralDater uses state-of-the-art causal and temporal relation extraction algorithm (Mirza and Tonelli, 2016) for extracting temporal graph for each document. A GCN is then run over this temporal graph to refine the embeddings from previous layer. We refer to this GCN as **Temporal GCN** or **T-GCN**. In this step, a special DCT node is introduced whose embedding is also learned by the T-GCN.

- **Classifier**: Embedding of the DCT node along with average pooled embeddings learned by S-GCN are fed to a fully connected softmax classifier which makes the final prediction about the date of the document.

Even though the previous discussion is presented in a sequential manner, the whole network is trained in a joint end-to-end manner using back-propagation.

5 NeuralDater Details

In this section, we present detailed description of various components of NeuralDater.

5.1 Context Embedding (Bi-LSTM)

Let us consider a document D with n tokens $w_1, w_2, ..., w_n$. We first represent each token by a k-dimensional word embedding. For the experiments in this paper, we use GloVe (Pennington et al., 2014) embeddings. These token embeddings are stacked together to get the document representation $\mathcal{X} \in \mathbb{R}^{n \times k}$. We then employ a Bi-directional LSTM (Bi-LSTM) (Hochreiter and Schmidhuber, 1997) on the input matrix $\mathcal{X}$ to obtain contextual embedding for each token. After stacking contextual embedding of all these tokens, we get the new document representation matrix $\mathcal{H}^{cntx} \in \mathbb{R}^{n \times r_{cntx}}$. In this new representation, each token is represented in a r_{cntx}-dimensional space. Our choice of LSTMs for learning contextual embeddings for tokens is motivated by the previous success of LSTMs in this task (Sutskever et al., 2014).

5.2 Syntactic Embedding (S-GCN)

While the Bi-LSTM is effective at capturing immediate local context of a token, it may not be as effective in capturing longer range dependencies among words in a sentence. For example, in Figure 1, we would like the embedding of token *approved* to be directly affected by *govt*, even though they are not immediate neighbors. A dependency parse may be used to capture such longer-range connections. In fact, similar features were exploited by (Chambers, 2012) for the document dating problem. NeuralDater captures such longer-range information by using another GCN run over the syntactic structure of the document. We describe this in detail below.

The context embedding, $\mathcal{H}^{cntx} \in \mathbb{R}^{n \times r_{cntx}}$ learned in the previous step is used as input to this layer. For a given document, we first extract its syntactic dependency structure by applying the Stanford CoreNLP's dependency parser (Manning et al., 2014) on each sentence in the document individually. We now employ the Graph Convolution Network (GCN) over this dependency graph using the GCN formulation presented in Section 3.2. We call this GCN the Syntactic GCN or S-GCN, as mentioned in Section 4.

Since S-GCN operates over the dependency graph and uses Equation 2 for updating embeddings, the number of parameters in S-GCN is directly proportional to the number of dependency edge types. Stanford CoreNLP's dependency parser returns 55 different dependency edge types. This large number of edge types is going to significantly over-parameterize S-GCN, thereby increasing the possibility of overfitting. In order to address this, we use only three edge types in S-GCN. For each edge connecting nodes w_i and w_j in $\mathcal{E}'$ (see Equation 1), we determine its new type $L(w_i, w_j)$ as follows:

- $L(w_i, w_j) = \rightarrow$ if $(w_i, w_j, l(w_i, w_j)) \in \mathcal{E}'$, i.e., if the edge is an original dependency parse edge

- $L(w_i, w_j) = \leftarrow$ if $(w_i, w_j, l(w_i, w_j)^{-1}) \in \mathcal{E}'$, i.e., if the edges is an inverse edge

- $L(w_i, w_j) = \top$ if $(w_i, w_j, \top) \in \mathcal{E}'$, i.e., if the edge is a self-loop with $w_i = w_j$

S-GCN now estimates embedding $h^{syn}_{w_i} \in \mathbb{R}^{r_{syn}}$ for each token w_i in the document using the for-

mulation shown below.

$$h_{w_i}^{syn} = f\left(\sum_{w_j \in \mathcal{N}(w_i)} \left(W_{L(w_i,w_j)} h_{w_j}^{cntx} + b_{L(w_i,w_j)}\right)\right)$$

Please note S-GCN's use of the new edge types $L(w_i, w_j)$ above, instead of the $l(w_i, w_j)$ types used in Equation 2. By stacking embeddings for all the tokens together, we get the new embedding matrix $\mathcal{H}^{syn} \in \mathbb{R}^{n \times r_{syn}}$ representing the document.

AveragePooling: We obtain an embedding h_D^{avg} for the whole document by average pooling of every token representation.

$$h_D^{avg} = \frac{1}{n} \sum_{i=1}^{n} h_{w_i}^{syn}. \quad (3)$$

5.3 Temporal Embedding (T-GCN)

In this layer, NeuralDater exploits temporal structure of the document to learn an embedding for the Document Creation Time (DCT) node of the document. First, we describe the construction of temporal graph, followed by GCN-based embedding learning over this graph.

Temporal Graph Construction: NeuralDater uses Stanford's SUTime tagger (Chang and Manning, 2012) for date normalization and the event extraction classifier of (Chambers et al., 2014) for event detection. The annotated document is then passed to CATENA (Mirza and Tonelli, 2016), current state-of-the-art temporal and causal relation extraction algorithm, to obtain a temporal graph for each document. Since our task is to predict the creation time of a given document, we supply DCT as unknown to CATENA. We hypothesize that the temporal relations extracted in absence of DCT are helpful for document dating and we indeed find this to be true, as shown in Section 7. Temporal graph is a directed graph, where nodes correspond to events, time mentions, and the Document Creation Time (DCT). Edges in this graph represent causal and temporal relationships between them. Each edge is attributed with a label representing the type of the temporal relation. CATENA outputs 9 different types of temporal relations, out of which we selected five types, viz., *AFTER*, *BEFORE*, *SAME*, *INCLUDES*, and *IS_INCLUDED*. The remaining four types were ignored as they were substantially infrequent.

Please note that the temporal graph may involve only a small number of tokens in the document.

Datasets	# Docs	Start Year	End Year
APW	675k	1995	2010
NYT	647k	1987	1996

Table 1: Details of datasets used. Please see Section 6 for details.

For example, in the temporal graph in Figure 2, there are a total of 5 nodes: two temporal expression nodes (*1995* and *four years after*), two event nodes (*adopted* and *approved*), and a special DCT node. This graph also consists of temporal relation edges such as (*four years after*, *approved*, *BEFORE*).

Temporal Graph Convolution: NeuralDater employs a GCN over the temporal graph constructed above. We refer to this GCN as the Temporal GCN or T-GCN, as mentioned in Section 4. T-GCN is based on the GCN formulation presented in Section 3.2. Unlike S-GCN, here we consider label and direction specific parameters as the temporal graph consists of only five types of edges.

Let n_T be the number of nodes in the temporal graph. Starting with $\mathcal{H}^{syn}$ (Section 5.2), T-GCN learns a r_{temp}-dimensional embedding for each node in the temporal graph. Stacking all these embeddings together, we get the embedding matrix $\mathcal{H}^{temp} \in \mathbb{R}^{n_T \times r_{temp}}$. T-GCN embeds the temporal constraints induced by the temporal graph in $h_{DCT}^{temp} \in \mathbb{R}^{r_{temp}}$, embedding of the DCT node of the document.

5.4 Classifier

Finally, the DCT embedding h_{DCT}^{temp} and average-pooled syntactic representation h_D^{avg} (see Equation 3) of document D are concatenated and fed to a fully connected feed forward network followed by a softmax. This allows the NeuralDater to exploit context, syntactic, and temporal structure of the document to predict the final document date y.

$$\begin{aligned} h_D^{avg+temp} &= [h_{DCT}^{temp}\,;\, h_D^{avg}] \\ p(y|D) &= \text{Softmax}(W \cdot h_D^{avg+temp} + b). \end{aligned}$$

6 Experimental Setup

Datasets: We experiment on Associated Press Worldstream (APW) and New York Times (NYT) sections of Gigaword corpus (Parker et al., 2011). The original dataset contains around 3 million

documents of APW and 2 million documents of NYT from span of multiple years. From both sections, we randomly sample around 650k documents while maintaining balance among years. Documents belonging to years with substantially fewer documents are omitted. Details of the dataset can be found in Table 1. For train, test and validation splits, the dataset was randomly divided in 80:10:10 ratio.

Evaluation Criteria: Given a document, the model needs to predict the year in which the document was published. We measure performance in terms of overall accuracy of the model.

Baselines: For evaluating NeuralDater, we compared against the following methods:

- **BurstySimDater** Kotsakos et al. (2014): This is a purely statistical method which uses lexical similarity and term burstiness (Lappas et al., 2009) for dating documents in arbitrary length time frame. For our experiments, we took the time frame length as 1 year. Please refer to (Kotsakos et al., 2014) for more details.

- **MaxEnt-Time-NER**: Maximum Entropy (MaxEnt) based classifier trained on hand-crafted temporal and Named Entity Recognizer (NER) based features. More details in (Chambers, 2012).

- **MaxEnt-Joint**: Refers to MaxEnt-Time-NER combined with year mention classifier as described in (Chambers, 2012).

- **MaxEnt-Uni-Time:** MaxEnt based discriminative model which takes bag-of-words representation of input document with normalized time expression as its features.

- **CNN:** A Convolution Neural Network (CNN) (LeCun et al., 1999) based text classification model proposed by (Kim, 2014), which attained state-of-the-art results in several domains.

- **NeuralDater**: Our proposed method, refer Section 4.

Hyperparameters: By default, edge gating (Section 3.3) is used in all GCNs. The parameter K represents the number of layers in T-GCN (Section 5.3). We use 300-dimensional GloVe embeddings and 128-dimensional hidden state for both GCNs and BiLSTM with 0.8 dropout. We used Adam (Kingma and Ba, 2014) with 0.001 learning rate for training.

Method	APW	NYT
BurstySimDater	45.9	38.5
MaxEnt-Time+NER	52.5	42.3
MaxEnt-Joint	52.5	42.5
MaxEnt-Uni-Time	57.5	50.5
CNN	56.3	50.4
NeuralDater	**64.1**	**58.9**

Table 2: Accuracies of different methods on APW and NYT datasets for the document dating problem (higher is better). NeuralDater significantly outperforms all other competitive baselines. This is our main result. Please see Section 7.1 for more details.

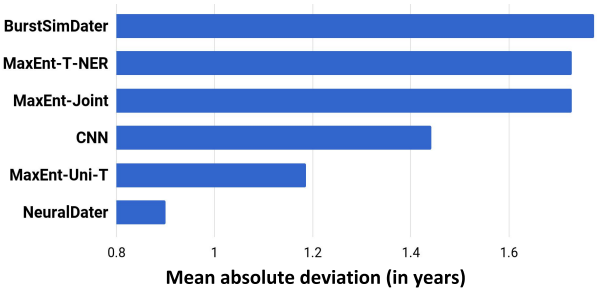

Figure 3: Mean absolute deviation (in years; lower is better) between a model's top prediction and the true year in the APW dataset. We find that NeuralDater, the proposed method, achieves the least deviation. Please see Section 7.1 for details.

Method	Accuracy
T-GCN	57.3
S-GCN + T-GCN ($K = 1$)	57.8
S-GCN + T-GCN ($K = 2$)	58.8
S-GCN + T-GCN ($K = 3$)	**59.1**
Bi-LSTM	58.6
Bi-LSTM + CNN	59.0
Bi-LSTM + T-GCN	60.5
Bi-LSTM + S-GCN + T-GCN (no gate)	62.7
Bi-LSTM + S-GCN + T-GCN ($K = 1$)	**64.1**
Bi-LSTM + S-GCN + T-GCN ($K = 2$)	63.8
Bi-LSTM + S-GCN + T-GCN ($K = 3$)	63.3

Table 3: Accuracies of different ablated methods on the APW dataset. Overall, we observe that incorporation of context (Bi-LSTM), syntactic structure (S-GCN) and temporal structure (T-GCN) in NeuralDater achieves the best performance. Please see Section 7.1 for details.

7 Results

7.1 Performance Comparison

In order to evaluate the effectiveness of NeuralDater, our proposed method, we compare it

against existing document dating systems and text classification models. The final results are summarized in Table 2. Overall, we find that NeuralDater outperforms all other methods with a significant margin on both datasets. Compared to the previous state-of-the-art in document dating, BurstySimDater (Kotsakos et al., 2014), we get 19% average absolute improvement in accuracy across both datasets. We observe only a slight gain in the performance of MaxEnt-based model (MaxEnt-Time+NER) of (Chambers, 2012) on combining with temporal constraint reasoner (MaxEnt-Joint). This may be attributed to the fact that the model utilizes only year mentions in the document, thus ignoring other relevant signals which might be relevant to the task. BurstySimDater performs considerably better in terms of precision compared to the other baselines, although it significantly underperforms in accuracy. We note that NeuralDater outperforms all these prior models both in terms of precision and accuracy. We find that even generic deep-learning based text classification models, such as CNN (Kim, 2014), are quite effective for the problem. However, since such a model doesn't give specific attention to temporal features in the document, its performance remains limited. From Figure 3, we observe that NeuralDater's top prediction achieves on average the lowest deviation from the true year.

7.2 Ablation Comparisons

For demonstrating the efficacy of GCNs and BiLSTM for the problem, we evaluate different ablated variants of NeuralDater on the APW dataset. Specifically, we validate the importance of using syntactic and temporal GCNs and the effect of eliminating BiLSTM from the model. Overall results are summarized in Table 3. The first block of rows in the table corresponds to the case when BiLSTM layer is excluded from NeuralDater, while the second block denotes the case when BiLSTM is included. We also experiment with multiple stacked layers of T-GCN (denoted by K) to observe its effect on the performance of the model.

We observe that embeddings from Syntactic GCN (S-GCN) are much better than plain GloVe embeddings for T-GCN as S-GCN encodes the syntactic neighborhood information in event and time embeddings which makes them more relevant for document dating task.

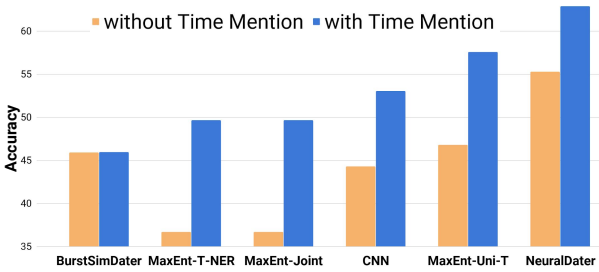

Figure 4: Evaluating performance of different methods on dating documents with and without time mentions. Please see Section 7.3 for details.

Overall, we observe that including BiLSTM in the model improves performance significantly. Single BiLSTM model outperforms all the models listed in the first block of Table 3. Also, some gain in performance is observed on increasing the number of T-GCN layers (K) in absence of BiLSTM, although the same does not follow when BiLSTM is included in the model. This observation is consistent with (Marcheggiani and Titov, 2017), as multiple GCN layers become redundant in the presence of BiLSTM. We also find that eliminating edge gating from our best model deteriorates its overall performance.

In summary, these results validate our thesis that joint incorporation of syntactic and temporal structure of a document in NeuralDater results in improved performance.

7.3 Discussion and Error Analysis

In this section, we list some of our observations while trying to identify pros and cons of NeuralDater, our proposed method. We divided the development split of the APW dataset into two sets – those with and without any mention of time expressions (year). We apply NeuralDater and other methods to these two sets of documents and report accuracies in Figure 4. We find that overall, NeuralDater performs better in comparison to the existing baselines in both scenarios. Even though the performance of NeuralDater degrades in the absence of time mentions, its performance is still the best relatively. Based on other analysis, we find that NeuralDater fails to identify timestamp of documents reporting local infrequent incidents without explicit time mention. NeuralDater becomes confused in the presence of multiple misleading time mentions; it also loses out on documents discussing events which are outside the time range of the text on which the model was trained. In future, we plan to eliminate these pitfalls by

incorporating additional signals from Knowledge Graphs about entities mentioned in the document. We also plan to utilize free text temporal expression (Kuzey et al., 2016) in documents for improving performance on this problem.

8 Conclusion

We propose NeuralDater, a Graph Convolutional Network (GCN) based method for document dating which exploits syntactic and temporal structures in the document in a principled way. To the best of our knowledge, this is the first application of deep learning techniques for the problem of document dating. Through extensive experiments on real-world datasets, we demonstrate the effectiveness of NeuralDater over existing state-of-the-art approaches. We are hopeful that the representation learning techniques explored in this paper will inspire further development and adoption of such techniques in the temporal information processing research community.

Acknowledgements

We thank the anonymous reviewers for their constructive comments. This work is supported in part by the Ministry of Human Resource Development (Government of India) and by a gift from Google.

References

James Allan, Ron Papka, and Victor Lavrenko. 1998. On-line new event detection and tracking. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, New York, NY, USA, SIGIR '98, pages 37–45. https://doi.org/10.1145/290941.290954.

Joost Bastings, Ivan Titov, Wilker Aziz, Diego Marcheggiani, and Khalil Simaan. 2017a. Graph convolutional encoders for syntax-aware neural machine translation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Copenhagen, Denmark, pages 1957–1967. https://www.aclweb.org/anthology/D17-1209.

Joost Bastings, Ivan Titov, Wilker Aziz, Diego Marcheggiani, and Khalil Sima'an. 2017b. Graph convolutional encoders for syntax-aware neural machine translation. *CoRR* abs/1704.04675. http://arxiv.org/abs/1704.04675.

Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann Lecun. 2014. Spectral networks and locally connected networks on graphs. In *International Conference on Learning Representations (ICLR2014), CBLS, April 2014*.

Nathanael Chambers. 2012. Labeling documents with timestamps: Learning from their time expressions. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers - Volume 1*. Association for Computational Linguistics, Stroudsburg, PA, USA, ACL '12, pages 98–106. http://dl.acm.org/citation.cfm?id=2390524.2390539.

Nathanael Chambers, Taylor Cassidy, Bill McDowell, and Steven Bethard. 2014. Dense event ordering with a multi-pass architecture. *Transactions of the Association of Computational Linguistics* 2:273–284. http://www.aclweb.org/anthology/Q14-1022.

Nathanael Chambers and Dan Jurafsky. 2008. Jointly combining implicit constraints improves temporal ordering. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Stroudsburg, PA, USA, EMNLP '08, pages 698–706. http://dl.acm.org/citation.cfm?id=1613715.1613803.

Nathanael Chambers, Shan Wang, and Dan Jurafsky. 2007. Classifying temporal relations between events. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*. Association for Computational Linguistics, Stroudsburg, PA, USA, ACL '07, pages 173–176. http://dl.acm.org/citation.cfm?id=1557769.1557820.

Angel X. Chang and Christopher Manning. 2012. Sutime: A library for recognizing and normalizing time expressions. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC-2012)*. European Language Resources Association (ELRA). http://www.aclweb.org/anthology/L12-1122.

Wisam Dakka, Luis Gravano, and Panagiotis G. Ipeirotis. 2008. Answering general time sensitive queries. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management*. ACM, New York, NY, USA, CIKM '08, pages 1437–1438. https://doi.org/10.1145/1458082.1458320.

Franciska M.G. de Jong, H. Rode, and Djoerd Hiemstra. 2005a. *Temporal Language Models for the Disclosure of Historical Text*, KNAW, pages 161–168. Imported from EWI/DB PMS [db-utwente:inpr:0000003683].

Franciska M.G. de Jong, H. Rode, and Djoerd Hiemstra. 2005b. *Temporal Language Models for the Disclosure of Historical Text*, KNAW, pages 161–168. Imported from EWI/DB PMS [db-utwente:inpr:0000003683].

Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. 2016. Convolutional neural networks on graphs with fast localized spectral filtering. In

Proceedings of the 30th International Conference on Neural Information Processing Systems. Curran Associates Inc., USA, NIPS'16, pages 3844–3852. http://dl.acm.org/citation.cfm?id=3157382.3157527.

Jennifer D'Souza and Vincent Ng. 2013. Classifying temporal relations with rich linguistic knowledge. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, pages 918–927. http://www.aclweb.org/anthology/N13-1112.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning*. MIT Press. `http://www.deeplearningbook.org`.

G. Hinton, L. Deng, D. Yu, G. E. Dahl, A. r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, and B. Kingsbury. 2012. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine* 29(6):82–97. https://doi.org/10.1109/MSP.2012.2205597.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Comput.* 9(8):1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735.

Nattiya Kanhabua and Kjetil Nørvåg. 2008a. Improving temporal language models for determining time of non-timestamped documents. In *Proceedings of the 12th European Conference on Research and Advanced Technology for Digital Libraries*. Springer-Verlag, Berlin, Heidelberg, ECDL '08, pages 358–370.

Nattiya Kanhabua and Kjetil Nørvåg. 2008b. Improving temporal language models for determining time of non-timestamped documents. In *International Conference on Theory and Practice of Digital Libraries*. Springer, pages 358–370.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, pages 1746–1751. https://doi.org/10.3115/v1/D14-1181.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *CoRR* abs/1412.6980.

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)*.

Dimitrios Kotsakos, Theodoros Lappas, Dimitrios Kotzias, Dimitrios Gunopulos, Nattiya Kanhabua, and Kjetil Nørvåg. 2014. A burstiness-aware approach for document dating. In *Proceedings of the 37th International ACM SIGIR Conference on Research & Development in Information Retrieval*. ACM, New York, NY, USA, SIGIR '14, pages 1003–1006. https://doi.org/10.1145/2600428.2609495.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1*. Curran Associates Inc., USA, NIPS'12, pages 1097–1105. http://dl.acm.org/citation.cfm?id=2999134.2999257.

Erdal Kuzey, Vinay Setty, Jannik Strötgen, and Gerhard Weikum. 2016. As time goes by: Comprehensive tagging of textual phrases with temporal scopes. In *Proceedings of the 25th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, WWW '16, pages 915–925. https://doi.org/10.1145/2872427.2883055.

Theodoros Lappas, Benjamin Arai, Manolis Platakis, Dimitrios Kotsakos, and Dimitrios Gunopulos. 2009. On burstiness-aware search for document sequences. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, New York, NY, USA, KDD '09, pages 477–486. https://doi.org/10.1145/1557019.1557075.

Yann LeCun, Patrick Haffner, Léon Bottou, and Yoshua Bengio. 1999. Object recognition with gradient-based learning. In *Shape, Contour and Grouping in Computer Vision*. Springer-Verlag, London, UK, UK, pages 319–. http://dl.acm.org/citation.cfm?id=646469.691875.

Heeyoung Lee, Angel Chang, Yves Peirsman, Nathanael Chambers, Mihai Surdeanu, and Dan Jurafsky. 2013. Deterministic coreference resolution based on entity-centric, precision-ranked rules. *Comput. Linguist.* 39(4):885–916.

Xiaoyan Li and W. Bruce Croft. 2003. Time-based language models. In *Proceedings of the Twelfth International Conference on Information and Knowledge Management*. ACM, New York, NY, USA, CIKM '03, pages 469–475. https://doi.org/10.1145/956863.956951.

D. Llidó, R. Berlanga, and M. J. Aramburu. 2001. Extracting temporal references to assign document event-time periods*. In Heinrich C. Mayr, Jiri Lazansky, Gerald Quirchmayr, and Pavel Vogel, editors, *Database and Expert Systems Applications*. Springer Berlin Heidelberg, Berlin, Heidelberg, pages 62–71.

Hector Llorens, Nathanael Chambers, Naushad UzZaman, Nasrin Mostafazadeh, James Allen, and James Pustejovsky. 2015. Semeval-2015 task 5:

Qa tempeval-evaluating temporal information understanding with question answering. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*. pages 792–800.

Inderjeet Mani and George Wilson. 2000. Robust temporal processing of news. In *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*. Association for Computational Linguistics, Stroudsburg, PA, USA, ACL '00, pages 69–76. https://doi.org/10.3115/1075218.1075228.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Association for Computational Linguistics (ACL) System Demonstrations*. pages 55–60. http://www.aclweb.org/anthology/P/P14/P14-5010.

Diego Marcheggiani and Ivan Titov. 2017. Encoding sentences with graph convolutional networks for semantic role labeling. *CoRR* abs/1703.04826. http://arxiv.org/abs/1703.04826.

Paramita Mirza and Sara Tonelli. 2014. Classifying temporal relations with simple features. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, pages 308–317. https://doi.org/10.3115/v1/E14-1033.

Paramita Mirza and Sara Tonelli. 2016. Catena: Causal and temporal relation extraction from natural language texts. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*. The COLING 2016 Organizing Committee, pages 64–75. http://www.aclweb.org/anthology/C16-1007.

MA Olson, K Bostic, MI Seltzer, and DB Berkeley. 1999. Usenix annual technical conference, freenix track.

Robert Parker, David Graff, Junbo Kong, Ke Chen, and Kazuaki Maeda. 2011. English gigaword fifth edition ldc2011t07. dvd. *Philadelphia: Linguistic Data Consortium* .

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*. pages 1532–1543. http://www.aclweb.org/anthology/D14-1162.

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

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*. MIT Press, Cambridge, MA, USA, NIPS'14, pages 3104–3112. http://dl.acm.org/citation.cfm?id=2969033.2969173.

Naushad UzZaman, Hector Llorens, Leon Derczynski, James Allen, Marc Verhagen, and James Pustejovsky. 2013. Semeval-2013 task 1: Tempeval-3: Evaluating time expressions, events, and temporal relations. In *Second Joint Conference on Lexical and Computational Semantics (* SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*. volume 2, pages 1–9.

Marc Verhagen, Robert Gaizauskas, Frank Schilder, Mark Hepple, Graham Katz, and James Pustejovsky. 2007. Semeval-2007 task 15: Tempeval temporal relation identification. In *Proceedings of the 4th international workshop on semantic evaluations*. Association for Computational Linguistics, pages 75–80.

Marc Verhagen, Roser Sauri, Tommaso Caselli, and James Pustejovsky. 2010. Semeval-2010 task 13: Tempeval-2. In *Proceedings of the 5th international workshop on semantic evaluation*. Association for Computational Linguistics, pages 57–62.

Xiaojun Wan. 2007. Timedtextrank: Adding the temporal dimension to multi-document summarization. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, New York, NY, USA, SIGIR '07, pages 867–868. https://doi.org/10.1145/1277741.1277949.

Michihiro Yasunaga, Rui Zhang, Kshitijh Meelu, Ayush Pareek, Krishnan Srinivasan, and Dragomir R. Radev. 2017. Graph-based neural multi-document summarization. In *Proceedings of CoNLL-2017*. Association for Computational Linguistics.

Katsumasa Yoshikawa, Sebastian Riedel, Masayuki Asahara, and Yuji Matsumoto. 2009. Jointly identifying temporal relations with markov logic. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*. Association for Computational Linguistics, pages 405–413. http://www.aclweb.org/anthology/P09-1046.