T3N: Harnessing Text and Temporal Tree Network forRumor Detection on Twitter

by

Nikhil Pinnaparaju, Manish Gupta, Vasudeva Varma

Report No: IIIT/TR/2021/-1

Centre for Language Technologies Research Centre
International Institute of Information Technology
Hyderabad - 500 032, INDIA
May 2021

T^3N: Harnessing Text and Temporal Tree Network for Rumor Detection on Twitter

Nikhil Pinnaparaju, Manish Gupta[⋆], Vasudeva Varma

nikhil.pinnaparaju@research.iiit.ac.in,
{manish.gupta,vv}@iiit.ac.in
IIIT Hyderabad, India

Abstract. Social media platforms have democratized the publication process resulting into easy and viral propagation of information. However, spread of rumors via such media often results into undesired and extremely impactful political, economic, social, psychological and criminal consequences. Several manual as well as automated efforts have been undertaken in the past to solve this critical problem. Existing automated methods are text based, user credibility based or use signals from the tweet propagation tree. We aim at using the text, user, propagation tree and temporal information jointly for rumor detection on Twitter. This involves several challenges like how to handle text variations on Twitter, what signals from user profile could be useful, how to best encode the propagation tree information, and how to incorporate the temporal signal. Our novel architecture, T^3N (Text and Temporal Tree Network), leverages deep learning based architectures to encode text, user and tree information in a temporal-aware manner. Our extensive comparisons show that our proposed methods outperform the state-of-the-art techniques by ∼7 and ∼6 percent points respectively on two popular benchmark datasets, and also lead to better early detection results.

1 Introduction

Social media portals provide a rich platform to share, forward, vote and review to encourage users to discuss online news. While this allows for faster and democratized publication of news, it also in-turn allows for malicious users to spread misinformation. Misinformation events have had immense economic impact in the past as evidenced by $130B stock market fluctuation due to "Barack Obama injured in explosion" rumor in 2017 [24], $4B drop in Apple market capitalization due to the "iPhone and Leopard delay email" rumor in 2008 [14], etc. Misinformation can even have criminal consequences. E.g., in 2016, a man was arrested after he walked into a popular pizza restaurant in Northwest Washington carrying an assault rifle and fired one or more shots. The man told police he had come to the restaurant to "self-investigate" an election-related rumor ("Mrs. Clinton is kidnapping, molesting and trafficking children in the back rooms of a D.C. pizzeria") that spread online during her presidential campaign. Lastly, spreading rumor and sharing misinformation undermines your credibility; in the past, news reporters have been suspended or fired for sharing misinformation [28].

[⋆] The author is also a Principal Applied Scientist at Microsoft.

Manual credibility verification on websites like Snopes, Politifact, etc. is challenging and time consuming. Automatic detection of misinformation from social media posts (especially tweets) is also challenging because (1) They are very short and do not typically follows English grammar rules. (2) Even real news appears on Twitter much faster than other news media like TV new channels, which means that it is difficult to cross-check such news with other sources. (3) Since these posts pertain to very fresh news, it is difficult to fact check against relatively stale information in knowledge bases. (4) Difficult to handle text variations on Twitter, identify best way to encode the tweet propagation tree information, and incorporate the temporal signal. Lastly, it is very important to detect such misinformation early, which makes the task arduous.

Previous work on rumor detection has mainly leveraged the post content or the network structure [38]. Content based methods have focused on (1) matching facts extracted from content with knowledge base facts, or (2) textual feature engineering [2, 31], or (3) image information [13]. However, textual representations prove to be insufficient for platforms where the amount of characters and therefore words is limited. Additionally, they rely on a very quickly updated knowledge base which is difficult. Hence, recently, network based approaches have been proposed. Such approaches are broadly of two types: propagation tree based [21, 33, 36] and user credibility based [9, 12, 27]. Overall, previous work has studied the importance of individual factors for predicting the credibility of social media posts.

We aim to study such factors jointly. We extract semantics from the post text content using deep recurrent models as well as Transformer models. We extract multiple features from the user profile. We combine such features along with the propagation tree structure to learn a semantic representation of the tree. Further, we learn representations of multiple temporal snapshots of the propagation tree. Representations of such snapshots are combined using another recurrent network to obtain a representation for the temporal tree. The text representation and the temporal tree representation are combined to finally predict credibility of the social media post. Figure 1 shows the architecture of our proposed system, T^3N (Text and Temporal Tree Network).

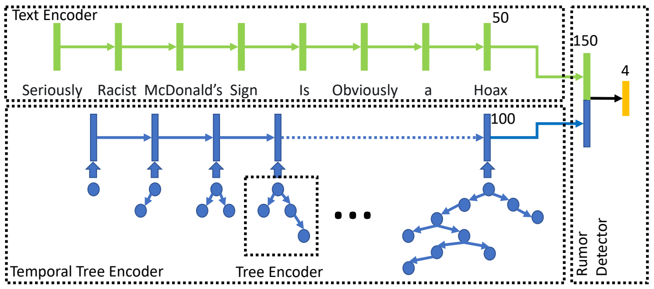

Fig. 1. T^3N System Architecture. T^3N could use a variety of text and tree encoders as discussed in Section 3. Here we show an instance with LSTM text encoder and temporal tree encoder.

.

Overall, in this paper, we make the following main contributions. (1) We propose to use the text content, propagation tree, user profile as well as temporal signals for early misinformation detection on Twitter. (2) We propose a novel deep learning architecture,

T^3N (Text and Temporal Tree Network), which consists of a text encoder and a temporal tree encoder to jointly model the above signals. (3) Our experiments with Twitter-15 and Twitter-16 benchmark datasets show that T^3N outperforms state-of-the-art methods by $\sim$7 and $\sim$6 percentage points respectively.

2 Related Work

Content based rumor detection: Content based methods have focused on (1) matching facts extracted from content with knowledge base facts [10], (2) textual feature engineering for style-based misinformation detection [2, 3, 22, 23]. Fact checking can be manual or automated. Popular manual fact checking websites include Snopes, Politifact, FactCheck, HoaxSlayer, TruthOrFiction, etc. Accuracy of automated fact checking [10, 26] is rather limited because they rely on a very quickly updated knowledge base for factual checking. While early studies used linguistic features [2, 23], more recently, deep neural networks (DNNs) [3, 21] have been explored for rumor detection. Content based representations prove to be insufficient for platforms where the amount of characters and therefore words is limited. Hence, our approach combines text with tree based signals.

Rumor detection using propagation trees: To address drawbacks of pure content based methods (following the observations made by Vosoughi et al. [32]), recently, propagation tree based methods [17, 20, 21, 33] have been proposed. Earlier efforts used graph kernels [20, 33] to capture high-order patterns differentiating different types of rumors by evaluating the similarities between their propagation tree structures. Recent DNNs like recursive neural networks (RNNs) [21], long short term memory (LSTM) networks [34], combination of recurrent and convolutional networks [17] have been explored. While it is beneficial to use propagation tree signals, it is important to carefully choose the right tree representation. We explore temporal trees.

Network based rumor detection: Apart from the simple propagation trees with only user nodes, rumor detection community has also studied other richer networks – both homogeneous [12, 25] as well as heterogeneous [9, 12, 27, 36]. Heterogeneous network based rumor detection is orthogonal to the direction we follow in this work. This is mainly due to lack of benchmark heterogeneous network datasets. We plan to explore whether this line of work complements our findings, as part of future work.

Fake profile detection on Twitter: As another relevant but orthogonal line of work, previous studies have attempted automated fake profile detection on Twitter. There are broadly two types of work in this area: (1) methods that analyze user behavior patterns [4] or content posted [6] or both [1], and (2) methods which jointly learn the user credibility along with credibility of other node types like tweets, events, etc.

3 T^3N: A System for Rumor Detection on Twitter

3.1 Problem Definition and T^3N System Overview

Both the benchmark Twitter-15 [16] and Twitter-16 [18] datasets have instances consisting of an origin tweet, its propagation tree and a class label. Each propagation tree is represented using tree edges which link a tweet to its reply or retweet. Each node (tweet) in the tree is represented by its userid, tweet id, and post time delay (in minutes). The class label is one of the following four classes: true news, rumor, debunking of rumor and unverified news[1]. Given this dataset, we model the problem of rumor detection as a 4-class classification problem. The classifier should take a new origin tweet with its propagation tree as input and output the appropriate class label with high accuracy.

We propose a novel deep text and temporal tree encoder network (T^3N) classifier. The basic idea behind T^3N is to jointly learn a unified representation of both the tweet content as well as the tweet's propagation tree as shown in Figure 1. It has the following four main components: (1) Text Encoder: It encodes the information from the tweet text into a latent vector. (2) Tree Encoder: It encodes the information from a propagation tree snapshot into a latent vector. (3) Temporal Tree Encoder: It encodes the information from a series of historical snapshots of the propagation tree into a latent vector. (4) Rumor Detector: It uses the learned combined text + temporal tree representation (latent vector) to predict one of the four classes. We discuss these components in detail in the remainder of this section.

3.2 Text Encoder

We use two kinds of text encoder architectures: recurrent architecture and Transformers.

RNNs like Gated Recurrent Units (GRUs) [5], LSTMs [11], bi-directional LSTMs [8] and their attention based variants have been found to be very effective in modeling text sequences across a large variety of natural language processing tasks mainly due to their representational power and effectiveness at capturing long-term dependencies. Attention allows the model to give more importance to certain set of words in the tweet while ignoring the others, effectively learning the focus points to better predict the correct rumor class for the tweet. The resultant tweet embedding, v_i, learned using any of these models, which is jointly learned during the training process captures the essential information from the tweet text. Fig. 1 shows an attention-based recurrent text encoder.

Another way of encoding text which has become popular in the past two years is using Transformer [30] based architectures like Bidirectional Encoder Representations from Transformers (BERT) [7]. The post text sequence is prepended with a "CLS" token. The representation C for the "CLS" token from the last encoder layer is used as the tweet text embedding. We also finetune the pre-trained model using labeled training data for the rumor prediction task.

3.3 Tree Encoder

Every propagation tree consists of the origin tweet as the root node. Replies and retweets of a tweet are represented using its children. Each node in the tree takes a user representation as input corresponding to the user who replied to or retweeted the parent tweet node, like in [17]. Tai et al. [29] extended regular linear LSTMs to child sum Tree-LSTMs. These allow for tree structured data (be it parse trees or propagation trees) to

[1] The label "debunking of rumor" denotes a news story that tells people that a certain news story is rumorous.

be trained without the loss of the data's inherent structure. While the standard LSTM composes its hidden state from the input at the current time step and the hidden state of the LSTM unit in the previous time step, the tree-structured LSTM, or Tree-LSTM, composes its state from an input vector and hidden states of arbitrarily many child units.

Given a tree, if we denote children of a node j by $C(j)$, the Child-Sum tree-LSTM transition equations can be written as follows.

$$\tilde{h}_j = \sum_{k \in C(j)} h_k; \;\; i_j = \sigma(W_i x_j + U_i \tilde{h}_j + b_i); o_j = \sigma(W_o x_j + U_o \tilde{h}_j + b_o) \tag{1}$$

$$f_{jk} = \sigma(W_f x_j + U_f h_k + b_f); \;\; c_j = i_j \circ tanh(W_c x_j + U_c \tilde{h}_j + b_c) + \sum_{k \in C(j)} f_{jk} \circ c_k \tag{2}$$

$$h_j = o_j \circ tanh(c_j) \tag{3}$$

Here, h_k and c_k denote the hidden layer and the cell state outputs of the k^{th} child respectively. x_j denotes the user profile vector for the node j. However, this method inspired by Tai et al. [29] does not handle temporal aspects of the propagation tree. Hence, we propose novel temporal-aware tree encoder methods in Sections 3.4 and 3.5.

3.4 Decayed Tree Encoder

Typically, the first few tweets in a rumor event are the most malicious in nature. Later tweets are often copies of the initial tweets. Also, intuitively, in a propagation tree for a rumor, the first few users (closer to the root of the tree) are malicious while users at lower levels (who reply to or retweet the original tweet) are naïve (who unintentionally engage in rumor propagation). Hence, we propose a modified Tree-LSTM variant that incorporates a temporal decay factor into our model. The variant provides less weight to propagation tree nodes that are sharing this misinformation at a later point of time. In some ways our time decay idea is also inspired by a similar notion as described in [20].

Each node k in the tree is associated with a timestamp $ts(k)$. We compute the decay factor corresponding to the node by comparing its timestamp with that of the original tweet (i.e, the root node). The modified Tree-LSTM transition equations can then be written as follows. Equations for $\tilde{h}_j$, i_j, o_j, c_j, h_j remain the same as in Eqs. 1 to 3.

$$td_k = ts(k) - ts(root); \;\; f_{jk} = \sigma(g(td_k) * (W_f x_j + U_f h_k + b_f)) \tag{4}$$

We experimented with two different forms of decay function g: (1) Linear: $g(td_k) = td_k$ (2) Exponential: $g(td_k) = e^{-td_k}$. Exponential decay function performed better compared to the linear function and hence we report results using exponential one.

3.5 Temporal Tree Encoder

We hypothesize that the order in which the nodes get added to a propagation tree depends on whether it contains true information or a rumor. Hence, we wish to learn a vector representation across multiple snapshots of the tree using a temporal tree encoder discussed as follows. Consider a tree with N nodes where each node has an associated timestamp. Such a tree can be represented as a time series of N snapshots

$S_1, S_2, ..., S_N$ where S_{i+1} has exactly one node ($(i+1)^{th}$) more than S_i and $(i+1)^{th}$ node has a timestamp larger than all of the nodes from 1 to i and smaller than all of the nodes from $i+2$ to N.

For each such tree snapshot, we use a tree encoder or a decayed tree encoder to get a latent representation. Parameters are shared across all the trees in the sequence. The temporal sequence of such latent representations across all snapshots S_1 to S_N are then combined using an LSTM. The last hidden layer output from the LSTM can then be considered as a latent representation for the temporal tree. Note that we use an LSTM rather than a BiLSTM because at test time, we will not have future tree snapshots.

If N is large, memory needed to learn the temporal tree representation could be large. Also, the number of computations needed are high. Hence, to generate the temporal tree representation we take $M+1$ equidistant snapshots from within the overall sequence, and then design an LSTM with just $M+1$ inputs. The gap between each unit in the LSTM input sequence is N/M. Note that M=1 is equivalent to learning from all snapshots of the temporal tree sequence. We experiment with different values of M and present results in Section 4.

3.6 Putting it all together

Overall, our proposed T^3N system first encodes the tweet text to obtain a text representation. Next, it creates a $M+1$-sized temporal tree sequence. Each tree is encoded using a tree encoder or a decayed tree encoder. Further, these individual $M+1$ tree representations are combined using a BiLSTM to obtain a temporal tree representation. Finally, the text and the temporal tree representations are concatenated, and connected to an output softmax layer with four neurons corresponding to the four class labels.

We use the cross entropy loss. The parameters of the text encoder, the temporal tree encoder and the misinformation detector are all initialized randomly and trained jointly using back propagation. We also experiment with separate pre-training and fine-tuning phases. In this way of training, we first train the text encoder using the labeled training data by directly connecting the output of the text encoder with an output layer in a fully connected manner. Next, we separately pre-train the tree encoder using the labeled training data by directly connecting the output of the temporal tree encoder with an output layer in a fully connected manner. Once the separate pre-training of text encoder and the temporal tree encoder is done, we use the end-to-end architecture shown in Figure 1 and fine-tune all the weights using the same labeled training data. We observed that such pre-training followed by fine-tuning provided better results compared to training end-to-end from scratch.

4 Experiments

4.1 Datasets, Experimental Settings and Baselines

Following the work in [2,15,17,19,20,21,33,35,37], we experiment with two popular benchmark datasets: Twitter-15 [16] and Twitter-16 [18] which are also the only two datasets that provide temporal network information along with the tweets. We eliminated (1) repeated edges from tree files (2) nodes that appear as children but appear before parent from the original datasets, and (3) nodes with negative post time delay). Table 1 shows statistics of the two datasets. Further Figs 2, 3, 4, 5 show distributions of important parameters of the two datasets. Note that the URLs in the tweets have been replaced by the keyword URL in the original dataset. The original dataset contains user IDs but not their profile information. Hence, we crawled user profiles using the Twitter API[2]. From these profiles we extract user profile attributes which could be correlated to their credibility. These features include length of the user profile description in words, length of username in characters, followers count, friends count, statuses count, registration age, whether the user is verified and whether the user is geo enabled. These features were also proposed in [17]. This feature vector of eight features is used as the data input vector for the tree-LSTMs in the tree encoder. The datasets can be downloaded from here[3].

Statistic	#stories	#true news	#rumors	#debunking of rumor	#unverified news	#users	#posts	Avg Tweet length	Avg Tree Depth	Avg #tree nodes
Twitter-15	1490	374	370	372	374	276663	331612	15.48	4.74	405.1
Twitter-16	818	205	205	207	201	173487	204820	15.19	4.89	428.9

Table 1. Statistics of the Twitter-15 and Twitter-16 Datasets

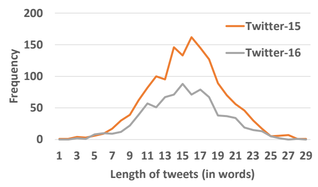

Fig. 2. Length distribution of tweets (in words)

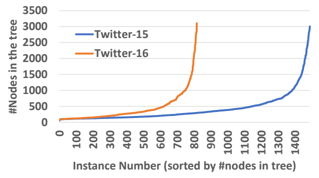

Fig. 3. Distribution of #nodes in the propagation tre

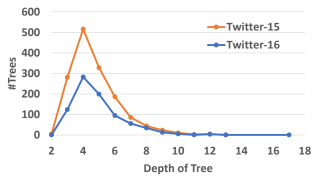

Fig. 4. Depth distribution of propagation trees

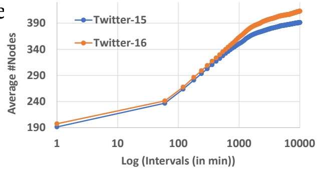

Fig. 5. Distribution of arrival time of nodes in the propagation trees (up to 1 week)

For our recurrent text encoders (GRUs and LSTMs), we used GloVe 100D embeddings and Adagrad optimizer with learning rate of 0.01. For BERT, we initialized using bert-base-uncased with standard 768D hidden size, and used AdamW optimizer with learning rate of 2e-5. No other specific pre-processing steps were performed. We train

[2] `https://developer.twitter.com/en/docs/accounts-and-users/follow-search-get-users/overview`
[3] `https://www.dropbox.com/s/7ewzdrbelpmrnxu/rumdetect2017.zip?dl=0`

all models for 10 epochs. For other settings for reproducibility, please refer to our code[4]. We perform 4-fold experiments and report the average.

Baselines: We compare with the following 9 previously proposed traditional machine learning methods: (1) SVM-BOW: Linear Support Vector Machines (SVM) with bag-of-words features extracted from the text in each tree. (2) DTC [2]: The information credibility model using a Decision-Tree Classifier based on various hand-crafted statistical features of the tweets. (3) SVM-RBF [35]: Same as (2) but uses SVM classifier with Radial Basis Function (RBF) kernel. (4) SVM-TS [19]: A linear SVM classification model that uses time-series to model the variation of a set of hand-crafted features. (5) DTR [37]: A Decision-Tree-based Ranking method to identify trending rumors. (6) RFC [15]: A Random Forest Classifier using three parameters to fit the temporal properties and an extensive set of hand-crafted features related to the user, linguistic and structure characteristics. (7) PTK [20]: Used kernel-based data-driven method called Propagation Tree Kernel (PTK) to generate relevant features (i.e., subtrees) automatically for estimating the similarity between two propagation trees. An SVM classifier is used on top of such features. (8) cPTK [20]: Extends PTK into a context-enriched PTK (cPTK) by considering different propagation paths from source tweet to the roots of subtrees, which capture the context of transmission. (9) SVM-HK [33]: An SVM classifier using features derived from propagation structures with Hybrid kernel.

Further, we compare with the following 5 previously proposed deep learning methods. (1) PPC_RNN+CNN [17]: Uses propagation path construction and transformation, RNN and CNN-based propagation path representation and propagation path classification. (2) PPC_RNN [17]: Same as PPC_RNN+CNN where CNN based representation is not used. (3) PPC_CNN [17]: Same as PPC_RNN+CNN where RNN based representation is not used. (4) BU-RvNN [21]: Uses a recursive neural model based on a bottom-up tree-structured neural networks for rumor representation learning and classification. (5) TD-RvNN [21]: Same as (4) but is top-down.

4.2 Accuracy Comparison

Results in Table 2 show that cPTK is the best traditional ML method across both the datasets. Multiple DL methods are better than the ML methods. Our proposed method, T^3N, outperforms all the other methods by a large margin. This particular variant of our system used BERT as the text encoder and Decay Temporal Tree encoder for encoding the propagation tree. Also, our method performs well across all the classes for both the datasets, except for the "rumor" class for Twitter-16 where the PPC_RNN+CNN is better. We believe this is because for Twitter-16, all of our text encoders perform relatively weakly for the "rumor" class (Table 3). On the other hand, this is not a concern for PPC_RNN+CNN since it uses a CNN also.

4.3 Ablation Studies

To understand the degree of contribution of various components towards rumor detection, we perform a series of ablation tests: using only text encoder, only tree encoder

[4] `https://www.dropbox.com/sh/nw14d4qd3zhm3mb/AAB843fUKQIXxVqsWnrRW5mfa?dl=0`

		Twitter-15					Twitter-16				
	Model	Acc.	T F1	R F1	D F1	U F1	Acc.	T F1	R F1	D F1	U F1
Traditional ML	SVM-BOW	0.548	0.564	0.524	0.582	0.512	0.585	0.553	0.556	0.655	0.578
	DTC	0.454	0.733	0.355	0.317	0.415	0.465	0.643	0.393	0.419	0.403
	SVM-RBF	0.318	0.455	0.037	0.218	0.225	0.321	0.423	0.085	0.419	0.037
	SVM-TS	0.544	0.796	0.472	0.404	0.483	0.574	0.755	0.420	0.571	0.526
	DTR	0.409	0.501	0.311	0.364	0.473	0.414	0.394	0.273	0.630	0.344
	RFC	0.565	0.810	0.422	0.401	0.543	0.585	0.752	0.415	0.547	0.563
	cPTK	0.750	0.804	0.698	0.765	0.733	0.732	0.740	0.709	0.836	0.686
	PTK	0.710	0.825	0.685	0.688	0.647	0.722	0.784	0.690	0.786	0.644
	SVM-HK	0.493	0.650	0.439	0.342	0.336	0.511	0.648	0.434	0.473	0.451
Deep learning	PPC_RNN	0.811	0.759	0.842	0.765	0.787	0.842	0.809	0.865	0.836	0.839
	PPC_CNN	0.803	0.737	0.835	0.751	0.775	0.847	0.812	0.871	0.833	0.841
	PPC_RNN+CNN	0.842	0.811	0.875	0.790	0.818	0.863	0.820	**0.898**	0.837	0.843
	BU-RvNN	0.708	0.695	0.728	0.759	0.653	0.718	0.723	0.712	0.779	0.659
	TD-RvNN	0.723	0.682	0.758	0.821	0.654	0.737	0.662	0.743	0.835	0.708
	Best T^3N	**0.912**	**0.905**	**0.915**	**0.912**	**0.914**	**0.927**	**0.957**	0.875	**0.970**	**0.909**

Table 2. Accuracy and class-wise F1 comparison across various methods for Twitter-15 and Twitter-16 datasets. Last row is our proposed method. (T=true, R=Rumor, D=debunk, U=unverified)

and multiple combinations of various text and tree encoders in Tables 3, 4 and 5 resp. In Table 3, we compare the accuracy obtained using multiple text encoders: GRU and LSTM, and their bi-directional variants. We also tried adding attention to these models, but it did not lead to significantly better results. We observe that bidirectional models do not lead to better results either. We also present the accuracy using BERT. Across both datasets, BERT clearly performs much better than any of the recurrent models.

	Twitter-15					Twitter-16				
Model	Acc.	T F1	R F1	D F1	U F1	Acc.	T F1	R F1	D F1	U F1
GRU	0.6040	0.6795	0.6438	0.5756	0.4894	0.6332	0.7086	0.5119	0.6870	0.4723
LSTM	0.5637	0.6881	0.5488	0.5445	0.4631	0.6528	0.7351	0.5169	0.6804	0.5127
Bi-GRU	0.5839	0.6833	0.6098	0.5659	0.4525	0.6235	0.7325	0.5146	0.6330	0.4508
Bi-LSTM	0.5496	0.6914	0.5430	0.5276	0.4223	0.6039	0.7448	0.4815	0.6018	0.4479
BERT	**0.7795**	**0.8342**	**0.7530**	**0.7554**	**0.7721**	**0.7744**	**0.8992**	**0.5553**	**0.7383**	**0.7278**

Table 3. Accuracy and class-wise F1 values for methods which use only tweet text information. (T=true, R=Rumor, D=debunk, U=unverified)

In Table 4, we compare the accuracy obtained using multiple tree encoders: Standard, Decay, Temporal, Decay Temporal. We also vary the number of trees as 1, 5, 10, 15 and 20. Overall, we observe that temporal tree representation gives better results compared to the standard one (where we just use the last snapshot of the tree). We also observe that as we increase the number of trees, typically the accuracy increases slightly. It is also important to note that the accuracy values obtained using tree only encoders are typically smaller than those using text only encoders. This could be because the text content has relatively much stronger signals compared to the tree.

In Table 5, we compare the accuracy obtained using multiple text and tree encoder combinations. For temporal and decay temporal tree encoders, we fixed the number of trees to 20. For standard tree encoder, of course, the number of trees is 1, i.e., the latest tree snapshot. As expected, results for text+tree methods (Table 5) are far better than results with just text (Table 3) or just tree (Table 4). The best combination is Decay Temporal tree encoder and BERT based text encoder. It is interesting to note that decay temporal usually performs better than temporal tree encoder in most cases. Further, we

Tree Encoder	# Trees	Twitter-15 Acc.	T F1	R F1	D F1	U F1	Twitter-16 Acc.	T F1	R F1	D F1	U F1
Standard	1	0.386	0.086	0.618	0.000	0.444	0.507	0.120	0.576	0.520	0.592
Decay	1	0.400	0.057	**0.664**	0.021	0.443	0.527	0.000	0.520	**0.587**	0.661
Temporal	5	0.437	0.475	0.610	0.378	0.203	0.522	0.077	0.537	0.544	0.649
Decay Temporal	5	0.458	0.462	0.621	0.376	0.358	0.571	0.290	0.644	0.579	0.640
Temporal	10	0.424	0.424	0.578	0.335	0.258	0.551	0.290	0.487	0.585	**0.671**
Decay Temporal	10	0.445	0.495	0.593	0.391	0.288	0.576	0.310	**0.673**	0.581	0.612
Temporal	15	0.464	**0.568**	0.605	0.352	0.113	0.517	0.246	0.546	0.496	0.639
Decay Temporal	15	**0.485**	0.514	0.372	0.541	0.540	0.551	0.478	0.500	0.520	0.644
Temporal	20	0.472	0.398	0.318	**0.587**	**0.578**	0.551	0.369	0.584	0.509	0.656
Decay Temporal	20	0.477	0.483	0.335	0.559	0.542	**0.576**	**0.500**	0.592	0.538	0.619

Table 4. Accuracy and class-wise F1 values for methods which use only propagation tree information. (T=true, R=Rumor, D=debunk, U=unverified)

Tree Encoder	Text Encoder	Twitter-15 Acc.	T F1	R F1	D F1	U F1	Twitter-16 Acc.	T F1	R F1	D F1	U F1
Standard	LSTM	0.625	0.699	0.549	0.559	0.710	0.629	0.745	0.547	0.633	0.596
Standard	GRU	0.625	0.701	0.568	0.597	0.652	0.659	0.764	0.598	0.674	0.621
Standard	BiLSTM	0.585	0.675	0.495	0.515	0.674	0.659	0.674	0.672	0.639	0.646
Standard	BiGRU	0.652	0.756	0.577	0.631	0.653	0.590	0.681	0.609	0.568	0.500
Standard	BERT	0.509	0.000	0.212	0.866	0.494	0.683	0.774	0.598	0.687	0.685
Temporal	LSTM	0.582	0.683	0.448	0.554	0.648	0.629	0.605	0.696	0.679	0.679
Temporal	GRU	0.619	0.696	0.561	0.570	0.647	0.683	0.725	0.631	0.685	0.703
Temporal	BiLSTM	0.595	0.573	0.467	0.688	0.679	0.600	0.651	0.569	0.489	0.684
Temporal	BiGRU	0.649	0.696	0.555	0.647	0.706	0.659	0.651	0.696	0.631	0.654
Temporal	BERT	0.895	0.913	0.899	0.876	0.902	0.751	0.860	0.905	0.571	0.667
Decay Temporal	LSTM	0.614	0.675	0.520	0.541	0.714	0.634	0.777	0.511	0.547	0.698
Decay Temporal	GRU	0.646	0.734	0.558	0.597	0.696	0.683	0.714	0.638	0.661	0.721
Decay Temporal	BiLSTM	0.619	0.674	0.570	0.591	0.639	0.663	0.667	0.633	0.626	0.726
Decay Temporal	BiGRU	0.614	0.651	0.532	0.590	0.688	0.644	0.607	0.696	0.661	0.600
Decay Temporal	BERT	**0.912**	**0.905**	**0.915**	**0.912**	**0.914**	**0.927**	**0.957**	**0.875**	**0.970**	**0.909**

Table 5. Accuracy and class-wise F1 values for various combinations of the text and tree encoders, i.e., variants of the T^3N system. (T=true, R=Rumor, D=debunk, U=unverified)

observed that the best model is confused the most between true and debunking of rumor classes. Intuitively, these classes are very similar and hence this is expected.

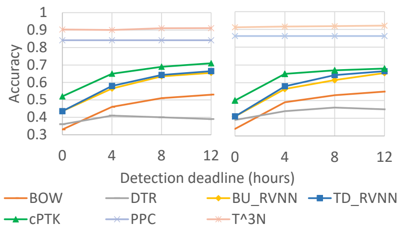

Fig. 6. Accuracy versus detection deadline (hours). Left: Twitter-15 and right: Twitter-16.

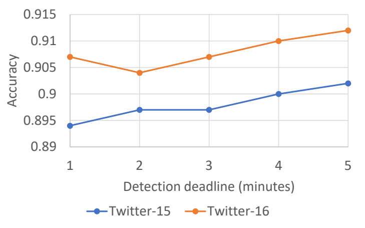

Fig. 7. Accuracy versus detection deadline for our best T^3N method (minutes).

4.4 Early Detection Results

Further, we wanted to check how early can we predict rumorous tweets. Fig. 6 shows the variation in accuracy wrt. time from original tweets in hours for various methods for the two datasets. We have ensured that we take the data points only until the time to detection. Note that our method shows an almost flat curve in Fig. 6 since it reaches a

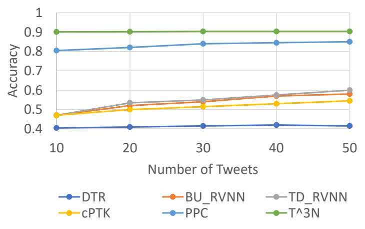

Fig. 8. Accuracy versus #tweets (Twitter-15).

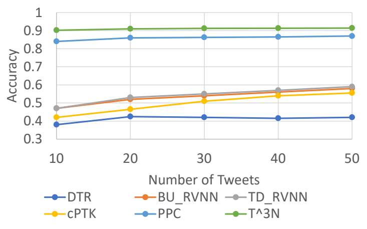

Fig. 9. Accuracy versus #tweets (Twitter-16).

high accuracy within the first 5 minutes of the original tweet itself (as shown in Fig. 7). We believe this is because of the best use of text, network and user profile data in our method. Although the accuracy using just the text encoder (at time=0) is relatively lower, the tree generated within 1–2 minutes is good enough for accuracy to be high. The best baseline is PPC. We also plot the variation in accuracy wrt. #tweets after the original tweet in Figs. 8 and 9 for the two datasets respectively. Again, we observe that our proposed method reaches high accuracy within a very few number of tweets.

Dataset	Tweet Text	Tree only classifier prediction	Text only classifier prediction	Actual Label	Text contribution in Best Model	Tree contribution in Best Model
Twitter-15	ca kkk grand wizard endorses @hillaryclinton #neverhillary #trump2016 URL	Unverified	Rumor	Unverified	0.820	0.180
Twitter-15	florida woman pays $20,000 for third breast — URL URL	Debunk	Rumor	Rumor	0.934	0.066
Twitter-15	one person dead, many taken to hospital after shootings, stabbing at denver coliseum, police say. URL	Rumor	Debunk	Debunk	0.922	0.078
Twitter-15	reporter charlo green quit on air to start a marijuana business URL will quitting on tv pay off? URL	Debunk	True	True	0.925	0.075
Twitter-16	chick-fil-a to open on sundays URL	Rumor	Unverified	Rumor	0.844	0.156
Twitter-16	dna confirms hakeem from ‘empire’ is jay-z’s biological son [read details URL URL]	Rumor	Debunk	Rumor	0.633	0.367
Twitter-16	72 dhs employees on terrorist watch list URL	Unverified	Rumor	Rumor	0.907	0.093
Twitter-16	#backtothefuture fans, it’s october 21, 2015 – the future is finally here! URL URL	Debunk	Rumor	Rumor	0.901	0.099
Twitter-16	#ripnathancirillo rt @ABC: soldier killed at war memorial identified as cpl. nathan cirillo #ottawashooting URL	Rumor	True	True	0.968	0.032

Table 6. Examples where the joint text+tree model provided correct prediction, and one of text-only or tree-only encoders provided correct prediction.

5 Conclusion

In this paper, we discussed the critical problem of rumor detection on Twitter. To the best of our knowledge this is the first work to explore the application of Transformer based models for this task. Besides using a BERT based text encoder, our system T^3N couples it with a temporal tree based encoder. Using two datasets and extensive comparisons with numerous previously proposed methods, we show the efficacy of our method.

References

1. Cai, C., Li, L., Zeng, D.: Detecting social bots by jointly modeling deep behavior and content information. In: CIKM. pp. 1995–1998 (2017)
2. Castillo, C., Mendoza, M., Poblete, B.: Information credibility on twitter. In: WWW. pp. 675–684 (2011)
3. Chen, T., Li, X., Yin, H., Zhang, J.: Call attention to rumors: Deep attention based recurrent neural networks for early rumor detection. In: PAKDD. pp. 40–52. Springer (2018)
4. Cheng, J., Bernstein, M., Danescu-Niculescu-Mizil, C., Leskovec, J.: Anyone can become a troll: Causes of trolling behavior in online discussions. In: CSCW. pp. 1217–1230 (2017)
5. Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., Bengio, Y.: Learning phrase representations using rnn encoder-decoder for statistical machine translation. arXiv:1406.1078 (2014)
6. Chu, Z., Gianvecchio, S., Wang, H., Jajodia, S.: Detecting automation of twitter accounts: Are you a human, bot, or cyborg? T. on Dependable and Secure Computing **9**(6), 811–824 (2012)
7. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv:1810.04805 (2018)
8. Graves, A., Jaitly, N., Mohamed, A.r.: Hybrid speech recognition with deep bidirectional lstm. In: Workshop on automatic speech recognition and understanding. pp. 273–278 (2013)
9. Gupta, M., Zhao, P., Han, J.: Evaluating event credibility on twitter. In: ICDM. pp. 153–164 (2012)
10. Hassan, N., Arslan, F., Li, C., Tremayne, M.: Toward automated fact-checking: Detecting check-worthy factual claims by claimbuster. In: KDD. pp. 1803–1812 (2017)
11. Hochreiter, S., Schmidhuber, J.: Long Short-Term Memory. Neural Computation **9**(8), 1735–1780 (1997)
12. Jin, Z., Cao, J., Zhang, Y., Luo, J.: News verification by exploiting conflicting social viewpoints in microblogs. In: AAAI (2016)
13. Khattar, D., Goud, J.S., Gupta, M., Varma, V.: Mvae: Multimodal variational autoencoder for fake news detection. In: WWW. pp. 2915–2921 (2019)
14. Krazit, T.: Engadget sends apple stock plunging on iphone rumor. CNET. com, May **16** (2007)
15. Kwon, S., Cha, M., Jung, K.: Rumor detection over varying time windows. PloS one **12**(1) (2017)
16. Liu, X., Nourbakhsh, A., Li, Q., Fang, R., Shah, S.: Real-time rumor debunking on twitter. In: CIKM. pp. 1867–1870 (2015)
17. Liu, Y., Wu, Y.F.B.: Early detection of fake news on social media through propagation path classification with recurrent and convolutional networks. In: AAAI. pp. 354–361 (2018)
18. Ma, J., Gao, W., Mitra, P., Kwon, S., Jansen, B.J., Wong, K.F., Meeyoung, C.: Detecting rumors from microblogs with recurrent neural networks. In: AAAI (2016)
19. Ma, J., Gao, W., Wei, Z., Lu, Y., Wong, K.F.: Detect rumors using time series of social context information on microblogging websites. In: CIKM. pp. 1751–1754 (2015)
20. Ma, J., Gao, W., Wong, K.F.: Detect rumors in microblog posts using propagation structure via kernel learning. In: ACL. pp. 708–717 (2017)
21. Ma, J., Gao, W., Wong, K.F.: Rumor detection on twitter with tree-structured recursive neural networks. In: ACL. pp. 1980–1989 (2018)
22. Ma, J., Gao, W., Wong, K.F.: Detect rumors on twitter by promoting information campaigns with generative adversarial learning. In: WWW. pp. 3049–3055. ACM (2019)
23. Popat, K., Mukherjee, S., Strötgen, J., Weikum, G.: Credibility assessment of textual claims on the web. In: CIKM. pp. 2173–2178 (2016)
24. Rapoza, K.: Can 'fake news' impact the stock market? by Forbes (2017)
25. Rath, B., Gao, W., Ma, J., Srivastava, J.: From retweet to believability: Utilizing trust to identify rumor spreaders on twitter. In: ASONAM. pp. 179–186. ACM (2017)
26. Shi, B., Weninger, T.: Discriminative predicate path mining for fact checking in knowledge graphs. Knowledge-based systems **104**, 123–133 (2016)
27. Shu, K., Wang, S., Liu, H.: Exploiting tri-relationship for fake news detection. arXiv:1712.07709 (2017)
28. Steel, E., Somaiya, R.: Brian williams suspended from nbc for 6 months without pay. The New York Times (2015)
29. Tai, K.S., Socher, R., Manning, C.D.: Improved semantic representations from tree-structured long short-term memory networks. In: ACL. pp. 1556–1566 (2015)
30. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. In: NIPS. pp. 5998–6008 (2017)
31. Volkova, S., Shaffer, K., Jang, J.Y., Hodas, N.: Separating facts from fiction: Linguistic models to classify suspicious and trusted news posts on twitter. In: ACL. pp. 647–653 (2017)
32. Vosoughi, S., Roy, D., Aral, S.: The spread of true and false news online. Science **359**(6380), 1146–1151 (2018)
33. Wu, K., Yang, S., Zhu, K.Q.: False rumors detection on sina weibo by propagation structures. In: ICDE. pp. 651–662 (2015)
34. Wu, L., Liu, H.: Tracing fake-news footprints: Characterizing social media messages by how they propagate. In: WSDM. pp. 637–645 (2018)
35. Yang, F., Liu, Y., Yu, X., Yang, M.: Automatic detection of rumor on sina weibo. In: KDD Workshop on Mining Data Semantics. pp. 1–7 (2012)
36. Zhang, J., Cui, L., Fu, Y., Gouza, F.B.: Fake news detection with deep diffusive network model. arXiv:1805.08751 (2018)
37. Zhao, Z., Resnick, P., Mei, Q.: Enquiring minds: Early detection of rumors in social media from enquiry posts. In: WWW. pp. 1395–1405 (2015)
38. Zhou, X., Zafarani, R.: Fake news: A survey of research, detection methods, and opportunities. arXiv:1812.00315 (2018)