Interpretable Rumor Detection in Microblogs by Attending to User Interactions

Ling Min Serena Khoo, Hai Leong Chieu
DSO National Laboratories
12 Science Park Drive
Singapore 118225
{klingmin,chaileon}@dso.org.sg

Zhong Qian, Jing Jiang
Singapore Management University
80 Stamford Road
Singapore 178902
qianzhongqz@163.com,jingjiang@smu.edu.sg

Abstract

We address rumor detection by learning to differentiate between the community's response to real and fake claims in microblogs. Existing state-of-the-art models are based on tree models that model conversational trees. However, in social media, a user posting a reply might be replying to the entire thread rather than to a specific user. We propose a post-level attention model (PLAN) to model long distance interactions between tweets with the multi-head attention mechanism in a transformer network. We investigated variants of this model: (1) a structure aware self-attention model (StA-PLAN) that incorporates tree structure information in the transformer network, and (2) a hierarchical token and post-level attention model (StA-HiTPLAN) that learns a sentence representation with token-level self-attention. To the best of our knowledge, we are the first to evaluate our models on two rumor detection data sets: the PHEME data set as well as the Twitter15 and Twitter16 data sets. We show that our best models outperform current state-of-the-art models for both data sets. Moreover, the attention mechanism allows us to explain rumor detection predictions at both token-level and post-level.

1 Introduction

The spread of fake news can have far reaching and devastating effects. $130 billion in stock value was wiped out in minutes after a false Associated Press's tweet claimed that Barack Obama was injured following an explosion in 2013 (**?**). Some researchers even believe that fake news has affected the outcome of the 2016 United States presidential election (**?**). The severity of the impact of fake news warrants the need for an effective and automated means of detecting fake news and has hence spurred much research in this area.

Our work focuses on the detection of fake claims using community response to such claims. This area of research exploits the collective wisdom of the community by applying natural language processing to comments directed towards a claim (see Figure 1). The key principle behind such works is that users on social media would share opinions, conjectures and evidences for inaccurate information. Hence, the interaction between users as well as the content shared could be captured for fake news detection. We discuss briefly two state-of-the-art models (**?**; **?**).

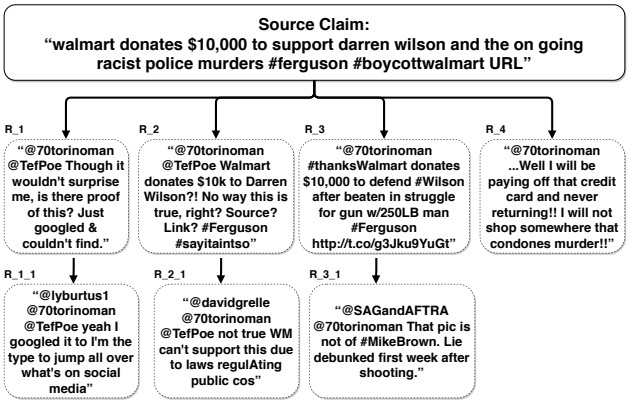


Figure 1: Sample of a thread from Twitter15 resulting from a fake claim.

Ma et al. (**?**) organized the source claim and its responding tweets in a tree structure as shown in Figure 1. Each tweet is represented as a node; the top node would be the source claim and the children of a node are tweets that have responded to it directly. They modeled the spread of information in a propagation tree using recursive neural networks. Signals from different nodes are aggregated recursively in either a bottom-up or a top-down manner. Information is propagated from the child node to the parent node in the bottom-up model and vice versa in a top-down model. Similarly, Kumar et al. (**?**) organized conversation threads with a tree structure and explored several variants of branch and tree LSTMs for rumour detection.

Both papers used tree models with the intention of modelling structural information present in the conversation thread. Information is propagated either from the parent to the child or vice versa in tree models. However, the thread structure in social media conversations might be unclear. Each user is often able to observe all the replies in different branches of the conversation. A user debunking a fake news may not be directed solely at the person he is replying to -

the content created could also be applicable to other tweets in the thread. Tree models do not model interactions between nodes from other branches explicitly and is a key limitation when modelling social media conversations. For example, in Figure 1, tweet *R_1* and its replying tweet, *R_1_1*, have expressed doubt regarding the factuality of the source claim. Tweets *R_2_1* and *R_3_1* have provided conclusive evidence that debunk the source claims as fake. Though *R_2_1* and *R_3_1* are child nodes of *R_2* and *R_3* respectively, they could provide important information to all the other nodes along the tree, such as *R_1_1* and *R_1*. Therefore, we should consider interactions between all tweets, not just those between parent and their child, for better aggregation of information from the replying tweets.

In this paper, we propose to overcome some limitations of tree models in modelling social media conversations as identified from our analysis. More specifically, we flattened the tree structure and arranged all tweets in a chronological order. We propose a post-level attention model (PLAN) that allows all possible pairwise interaction between tweets with the self-attention mechanism. To combine the strengths of both self-attention mechanism and tree models, we incorporated structural information and performed structure aware self-attention at a post-level (StA-PLAN). Lastly, we designed a structure aware hierarchical token and and post-level attention network (StA-HiTPLAN) to learn more complex sentence representations for each tweet.

The contributions of this paper are the following:

- We utilize the attention weights from our model to provide both token-level and post-level explanations behind the model's prediction. To the best of our knowledge, we are the first paper that has done this.
- We compare against previous works on two data sets - PHEME 5 events (**?**) and Twitter15 and Twitter16 (**?**). Previous works only evaluated on one of the two data sets.
- Our proposed models could outperform current state-of-the-art models for both data sets.

The rest of the paper is organized as follows. In Section 2, we examine some related work. We define our problem statement in Section 3. We present our models in Section 4 and results in Section 5. We then conclude with future work in Section 6.

2 Related Work

Existing approaches on automatically differentiating fake from real claims leverage on a variety of features: (i) the content of the claim, (ii) the bias and social network of the source of the claim, (iii) fact checking with trustworthy sources (e.g., Wikipedia), and (iv) community response to the claims. Our work falls into the last class of detecting rumors or fake claims from community response. In this section, we give a brief review of each class of work, focusing on works that detect fake claims using community response. For a more detailed survey, we refer the reader to (**?**).

Content Information: Early work on deceptive content detection studied the use of linguistic cues such as percentage of pronouns, word length, verb quantity, and word classes (**?**; **?**; **?**; **?**) on fake reviews, witness accounts, and satire. The detection of fake news using linguistic features have also been studied in (**?**; **?**). Such analysis of deceptive content relies on linguistic features which might be unique to domains or topics.

Source and Social Network: Another group of work studied the source of fake news, and its social network. Wang et al. (**?**) found that adding source information to the content improves fake news classification accuracy. Hu et al. (**?**) found that accounts created to spread fake news tend to have different social network characteristics.

Fact Checking: Fact checking websites such as politifact.com and snopes.com rely on manual verification to debunk fake news, but are unable to match up the rate at which fake news are being generated (**?**). Automated fact checking aims to check claims against trustworthy sources such as Wikipedia (**?**). More recently, Thorne et al. (**?**) proposed the FEVER shared task to verify an input claim against a database of 5 million Wikipedia documents, and classify each claim into one of three classes: *Supports*, *Refute* or *Not Enough Info*. Fact checking is a more principled approach to fake news detection. However, it also requires an established corpus of verified facts and may not work for new claims with little evidence.

Community Response: Detecting fake news from community response is most closely related to our paper. Researchers have worked on automatically predicting the veracity of a claim by building classifiers that leverages on the comments and replies to social media posts, as well as the propagation pattern (**?**; **?**; **?**; **?**). Ma et al. (**?**) adopted a multi-task learning approach to build a classifier that learns stance aware features for rumour detection. Similarly, Li et al. (**?**) adopted a multi-task learning approach for his models and included user information in his models. Chen et al. (**?**) proposed to pool out distinctive features that captures contextual variations of posts over time. Beyond linguistic features, other researchers have also looked at the demographics of the users (**?**; **?**) or interaction periodicity (**?**) to determine the credibility of users. Yang et al. (**?**) collected the user characteristics who were engaged in spreading fake news and used only user characteristics to build a fake news detector by classifying the propagation path. Li et al. (**?**) used a combination of user information and content features to train a LSTM with multi-task learning objective.

In this paper, we work on detecting rumor and fake news solely from the post and comments. Unlike **?**; **?** (**?**; **?**), we do not use the identities of user accounts. Most closely related to our work is Ma et al. (**?**; **?**; **?**) and Kumar et al. (**?**). Ma et al. (**?**; **?**; **?**) used recurrent neural network, propagation tree kernels and recursive neural trees to model posts in the same thread either sequentially or with a tree structure to predict whether the sequence pertains to rumour or real news. Kumar et al. (**?**) used various variants of LSTM (Branch LSTM, Tree LSTM and Binarized Constituency Tree LSTM). In this paper, instead of recursive tree models, we propose a transformer network for rumor detection.

3 Rumor Detection

In this section, we first define our problem statement. We address the problem of predicting the veracity of a claim given all of its responding tweets and the interactions between these tweets. We define each thread to be:

$$X = \{x_1, x_2, x_3, ..., x_n\},$$

where x_1 is the source tweet, x_i the i^{th} tweet in chronological order, and n the number of tweets in the thread.

Besides the textual information, there is also structural information in the conversation tree which could be exploited (**?**; **?**). In the tree-structured models, a pair of tweets x_i and x_j are only related if either x_i is replying to x_j or vice versa. In all of our proposed models, we allow any post to attend to any other post in the same thread. In our structure aware models, we label the relations between any pair of tweets x_i and x_j with a relation label, $R(i, j) \in \{$parent, child, before, after, self$\}$. The value of $R(i, j)$ is obtained by applying the following set of rules in sequence: **parent**: if x_i is directly replying x_j, **child**: if x_j is directly replying x_i, **before**: if x_i comes before x_j, **after**: if x_i comes after x_j, **self**: if $i = j$.

The rumor detection task reduces to learning to predict each (X, R) to its rumor class y. We conducted experiments on two rumor detection data sets, namely the Twitter15 and Twitter16 data, and the PHEME 5 data. The classes are different for the data sets we are working on:

- for Twitter15 and Twitter16: $y \in \{$*non-rumor*, *false-rumor*, *true-rumor*, *unverified*$\}$, and
- for PHEME, $y \in \{$*false-rumor*, *true-rumor*, *unverified*$\}$.

4 Approaches

For completeness, we first provide a brief description of recursive neural networks and the attention mechanism in transformer networks. We then describe the proposed models that is primarily based on the attention mechanism.

4.1 Recursive Neural Networks

Ma et al. (**?**) applied tree-structured recursive neural networks (RvNN) to the rumor detection problem: each thread is represented as a tree where the root is the claim, and each comment is a child to the post it is responding to. In RvNN, the input of each node in the tree is a vector of tf-idf values of the words in the post. Ma et al. (**?**) studied two models, a bottom-up and a top-down tree models. Kumar et al. (**?**) applied a similar mechanism.The formulation is appealing as we expect information from denying and questioning comments to be propagated and used for the classification of rumors.

However, as we will see in the data statistics in Table 1, the trees in the data sets are very shallow, with the majority of comments replying directly to the source tweet instead of replying to other tweets. We found that in social media, as often the entire thread is visible, a user replying to the root post might be continuing the conversation of earlier users and not specifically writing a reply to the root post. Tree models that do not explicitly model every possible pairwise interaction between tweets are therefore sub-optimal in modelling social media conversations. Therefore, we propose to overcome the limitations with our described models in subsequent sections.

4.2 Transformer Networks

The attention mechanism in transformer networks enable effective modeling of long-range dependencies (**?**). This is an advantage for rumor detection since there could be many replying tweets in a conversation and it is vital to be able to model the interactions between the tweets effectively. We briefly discuss the multi-head attention (MHA) layer in the transformer network and refer the reader to (**?**) for more details. Each MHA layer in the transformer network is made up of a self-attention sublayer and fully connected feed-forward sublayer. In the self-attention sublayer, a query and a set of key-value pair is mapped to an output. The output is a weighted sum of the values, where the weight for each value is determined by the compatibility between the query and the corresponding key. The compatibility, α_{ij}, is the attention weight between a query from position i and a key from position j, and is calculated with simple scaled dot-product attention:

$$\alpha_{ij} = \text{Compatibility}(q_i, k_j) = \text{softmax}(\frac{q_i k_j^T}{\sqrt{d_k}}) \quad (1)$$

The output at each position, z_i, would then be a weighted sum of the compatibility and value at other positions.

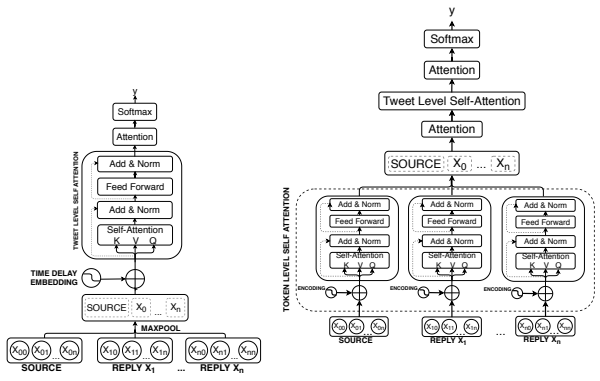
$$z_i = \sum_{j=1}^{n} \alpha_{ij} v_j, \quad (2)$$

where $\alpha_{ij} \in [0, 1]$ is the attention weight from Equation 1. A higher value indicates higher compatibility.

In order to allow the model to jointly attend to information from different representation subspaces at different positions, Vaswani et al. (**?**) introduced the concept of multi-head attention. Queries, keys and values are projected h times with different learned linear projections. Each projected versions of queries, keys and values would perform the attention function (Equation 2) in parallel, yielding h output values. These are concatenated and once again projected, generating the final values. The final layers would then be passed through a fully connected feed-forward sublayer consisting of two linear layers with a RELU activation between them.

4.3 Post-Level Attention Network (PLAN)

The architecture of our post-level attention network (PLAN) is shown in Figure 2a. We flattened the structure of the conversation tree and arranged the tweets chronologically in a linear structure with the source tweet as the first tweet. For our PLAN model, we applied max pooling to each tweet x_i in the linear structure to obtain it's sentence representation x'_i . We then pass a sequence of sentence embedding $X' = (x'_1, x'_2, ..., x'_n)$ through s number of multi-head attention (MHA) layers to model the interactions between the tweets. We refer to these MHA layers as post-level attention layers. As such, this transforms $X' = (x'_1, x'_2, ..., x'_n)$



(a) PLAN Model (b) StA-HiTPLAN Model

Figure 2: Proposed models

to $U = (u_1, u_2, ..., u_n)$. Lastly, we used the attention mechanism to interpolate the tweets before passing through a fully connected layer for prediction.

$$\alpha_k = \text{softmax}(\gamma^T u_k), \tag{3}$$

$$v = \sum_{k=0}^{m} \alpha_k u_k, \tag{4}$$

$$p = \text{softmax}(W_p^T v + b_p), \tag{5}$$

where $\gamma \in \mathbb{R}^{d_{model}}$, $\alpha_k \in \mathbb{R}$, $W_p \in \mathbb{R}^{d_{model},K}$, $b \in \mathbb{R}^{d_{model}}$, u_k is the output after passing through s number of MHA layers, K is the number of output classes, v and p are the representation vector and prediction vector for X respectively.

4.4 Structure Aware Post-Level Attention Network (StA-PLAN)

One possible limitation of our model is that we lose structural information by organising tweets in a linear structure. Structural information that are inherently present in a conversation tree might still be useful for fake news detection. Tree models are superior in this aspect since the structural information is modelled explicitly. To combine the strengths of tree models and the self-attention mechanism, we extended our PLAN model to include structural information explicitly. We adopted the formula in **?** (**?**) to perform structure aware self-attention:

$$\alpha_{ij} = \text{softmax}(\frac{q_i k_j^T + a_{ij}^K}{\sqrt{d_k}}), \tag{6}$$

$$z_i = \sum_{j=1}^{n} \alpha_{ij}(v_j + a_{ij}^V) \tag{7}$$

Equation 6 extends upon Equation 1 by explicitly adding a_{ij}^K when computing compatibility. Likewise, Equation 7 extends upon Equation 2 by adding a_{ij}^V when computing the output vector. Both a_{ij}^V and a_{ij}^K are vectors that represents one of the five possible structural relationship between the pair of the tweets (i.e. parent, child, before, after and self) as described in Section 3. These vectors are learned parameters in our model. We learn two distinct vectors to ensure the suitability for use of these vectors in the two different equations. Intuitively, a_{ij}^K gives the compatibility function more information to better determine compatibility; compatibility is now determined by both textual content and structural relationship of a pair of tweet instead of solely textual content. The addition of a_{ij}^V allows both structural and content information to be propagated to other tweets.

4.5 Structure Aware Hierarchical Token and Post-Level Attention Network (StA-HiTPLAN)

Our PLAN model performs max-pooling to get the sentence representation of each tweet. However, it could be more ideal to allow the model to learn the importance of the word vectors instead. Hence, we propose a hierarchical attention model - attention at a token-level then at a post-level. An overview of the hierarchical model is shown in Figure 2b.

Instead of using max-pooling to obtain sentence representation, we performed token-level self-attention before using the attention mechanism to interpolate the output. This approach would also learn a more complex sentence representation for each tweet. More formally, each tweet could be represented as a sequence of word tokens where $x_i = (x_{i,1}, x_{i,2}, ..., x_{i,|x_i|})$. We passed the sequence of word tokens in a tweet through s_{word} number of MHA layers. This allows interactions between tokens in a tweets and we refer to these layers as token-level attention layers. After which, we used the attention mechanism to interpolate the output from the MHA layers to obtain a sentence representation for each tweet. The sentence embedding for each tweet will then be used to perform structure aware post-level self-attention as described in Section 4.4.

4.6 Time Delay Embedding

Tweets created at different time intervals could be interpreted differently. Tweets expressing disbelief when a source claim is first created could be common as the claim may have not been verified. However, doubtful tweets at later stage of propagation could indicate a high tendency that the source claim is fake. Therefore, we investigated the usefulness of encoding tweets with time delay information with all three of our proposed models - PLAN, StA-PLAN and StA-HiTPLAN.

To include time delay information for each tweet, we bin the tweets based on their latency from the time the source tweet was created. We set the total number of time bins to be 100 and each bin represents a 10 minutes interval. Tweets with latency of more than 1,000 minutes would fall into the last time bin. We used the positional encoding formula introduced in the transformer network in **?** (**?**) to encode each time bin. The time delay embedding would be added to the sentence embedding of tweet. The time delay embedding, TDE, for each tweet is:

$$\text{TDE}_{pos,2i} = \sin \frac{pos}{10000^{2i/d_{model}}}, \tag{8}$$

$$\text{TDE}_{pos,2i+1} = \cos \frac{pos}{10000^{2i/d_{model}}}, \tag{9}$$

Data set	Twitter15	Twitter16	PHEME
Tree-depth	2.80	2.77	3.12
Num leaves	34.2	31.9	10.3
Num tweets	40.6	37.9	14.9
False	334	172	393
True	350	189	1008
Unverified	358	190	571
Non-rumor	371	205	-
Total trees	1413	756	1972
Total tweets	57,368	27,652	29,383

Table 1: Average tree depths, number of leaves and tweets.

where pos represents the time bin each tweet fall into and $pos \in [0, 100)$, i refers to the dimension and d_{model} refers to the total number of dimensions of the model.

5 Experiments and Results

We evaluate our model based on two rumour detection data sets - (i) Twitter15 and Twitter16 and (ii) PHEME 5 events data set. We show the statistics of the data sets in Table 1.

5.1 Data and Pre-processing

For the PHEME 5 data set, we follow the experimental setting of **?** (**?**) in using event wise cross-validation. For the Twitter15 and Twitter 16 data sets, there is a large proportion of retweets in each claim: 89% for Twitter15 and 90% for Twitter16. As we assume that retweets do not contribute new information to the model, we removed all retweets for Twitter15 and Twitter16. After the removal of retweets, we observe that a small number of claims would be left with only the source tweet. Since the principle behind our methodology is that we could exploit crowd signals for rumor detection, claims without any replies should then be "unverified". Hence, we amended the labels for such claims to be "unverified" in the training data (1.49% of Twitter15 - 8 False, 10 True and 3 Non-Rumour. 1.46% of Twitter16 - 9 False, 2 True and 0 Non-Rumour). In order for our results to be comparable with previous work, we excluded such claims from our testing set. We used the original splits released by (**?**) to split our data. We show the statistics of the data sets after pre-processing in Table 1.

5.2 Experimental Setup

In all experiments, we used the GLOVE 300d (**?**) embedding to represent each token in a tweet. (Preliminary experiments with BERT (**?**) did not improve results, and were computationally far more expensive than GLOVE). We used the same set of hyper parameters in all of our experiments for both PHEME and Twitter15 and Twitter16. Our model dimension is 300 and the dimension of intermediate output is 600. We used 12 post-level MHA layers and 2 token-level MHA layers. For training of the model, we used the ADAM optimizer with 6000 warm start-up steps. We used an initial learning rate of 0.01 with 0.3 dropout. We used a batch size of 32 for PLAN and StA-PLAN, and 16 for StA-HiTPLAN due to memory limitations of the GPUs used. We compare the following models:

- PLAN: The post-level attention network in Section 4.3.
- StA-PLAN: The structure aware post-level attention network in Section 4.4.
- StA-HiTPLAN: The structure aware hierarchical attention model in Section 4.5.

We also investigate these models with and without the time-delay embedding as described in Section 4.6. However, as time delay information did not improve upon PLAN and StA-PLAN for Twitter15 and Twitter16, we did not run experiments for StA-HiTPLAN + time delay for Twitter15 and Twitter16. For Twitter15 and Twitter 16, we compared our proposed models with RvNN models proposed by Ma et al. (**?**). As we are using different preprocessing steps (described in 5.1) for Twitter15 and Twitter16, we retrained the RvNN models with our implementation. We report both the original results and results from our implementation. For PHEME, we compare against the LSTM models proposed by Kumar et al. (**?**). As there were several combination of embeddings and models in (**?**), we compare our results with the best results reported for each variant of LSTM model proposed.

We summarize our experimental results in Tables 2 and 3. For Twitter 15 and Twitter 16, all our models outperform tree recursive neural networks, and our best models outperform by 14.2% for Twitter15 and 6.8% for Twitter16. For PHEME, our best model outperforms previous work by 1.6% F1-score.

5.3 Discussion of results

In the rest of this section, we analyze the performance of our proposed models on Twitter15, Twitter16 and PHEME. We found conflicting conclusions for the different datasets. In particular, although all of the datasets are similar in nature, the state-of-the-art performance of PHEME is far worse than that of Twitter15 and Twitter16. As such, we also provide analysis suggesting possible explanations for this disparity in results for the two datasets.

Structural Information: We proposed two methods - StA-PLAN and time-delay embeddings that aim to capture structural and chronological information among posts. StA-PLAN is the best performer for Twitter15 but did not outperform PLAN for Twitter16, though results were not substantially different. The reason structural aware model only works for Twitter15 might be because the Twitter15 data set is substantially bigger (in terms of number of tweets, see Table 1) than both PHEME and Twitter16. A big data set might be necessary to exploit the complicated structural information in the structure aware model. Time-delay information was useful for PHEME, where all proposed models performed better with time delay. On the other hand, time delay information was not useful for Twitter15 and Twitter16. Overall, it is unclear if structural information is useful for these data sets, and we leave further investigation to future work.

Token level self-attention: We proposed a token level self-attention mechanism to model the relationship between tokens in a tweet with our StA-HiTPLAN model. StA-

Method	Twitter15 Accuracy	F	T	U	NR	Twitter16 Accuracy	F	T	U	NR
BU-RvNN (Original)	70.8	72.8	75.9	65.3	69.5	71.8	71.2	77.9	65.9	72.3
TD-RvNN (Original)	72.3	75.8	82.1	65.4	68.2	73.7	74.3	83.5	70.8	66.2
BU-RvNN (Ours)	70.5	71.0	72.1	73.0	65.5	80.6	75.5	89.3	83.0	73.4
TD-RvNN (Ours)	65.9	66.1	68.9	71.4	55.9	76.7	69.8	87.2	81.3	66.1
PLAN	84.5	**85.8**	**89.5**	80.2	82.3	**87.4**	**83.9**	91.7	**88.8**	**85.3**
StA-PLAN	**85.2**	84.6	88.4	**83.7**	84.0	86.8	83.3	**92.7**	**88.8**	82.6
StA-HiTPLAN	80.8	80.2	85.1	76.0	81.7	80.7	76.5	88.8	82.0	74.9
PLAN + time-delay	84.1	84.2	87.3	80.3	84.2	84.8	77.6	89.7	85.6	84.9
StA-PLAN + time-delay	85.0	85.7	88.3	81.4	**84.4**	86.6	83.3	92.3	86.6	84.2

Table 2: Accuracy on Twitter15 and Twitter16, where F, T, U and NR stands for False, True, Unverified and Non-rumor respectively. We report the F1-Score for each individual class. The results of rows with (Original) were referenced from (**?**), while the remaining rows are based on our own implementation of the models.

Method	Macro F-Score
Branch LSTM - Multitask	35.9
Tree LSTM - Multitask	37.9
BCTree LSTM - Multitask	37.1
PLAN	36.0
StA-PLAN	34.9
StA-HiTPLAN	37.9
PLAN + Time Delay	38.6
StA-PLAN + Time Delay	36.9
StA-HiTPLAN + Time Delay	**39.5**
StA-HiTPLAN + Time Delay (Random split)	77.4

Table 3: F1-score on PHEME. We used the same train-test splits as (**?**) (Except the last row) and the results of the first three rows were referenced from the paper.

Words	Twitter15	Twitter16	PHEME
"*true*"	20.4	17.6	9.5
"*real*"	34.4	25.4	11.7
"*fake*"	14.1	10.9	2.2
"*lies*"	5.9	17.6	2.1

Table 4: Percentage of claims containing each word.

HiTPLAN was the best performer for PHEME but did not outperform our baseline model for Twitter15 and Twitter16.

To investigate why StA-HiTPLAN was the best performer for PHEME, we hypothesize that the signals for rumor detection in PHEME could be much weaker or expressed in a more implicit manner. Therefore, it would be necessary to study the interaction between the words to better capture the meaning of the whole sentence. To this end, we identified individual words that could be useful in determining veracity of the claim. We compute the statistics of the words "*true*", "*real*" , "*fake*" and "*lies*" in the three data sets as shown in Table 4. These words act as a proxy for crowd signals for the model to learn from and we observe that the usage of such words is the lowest in PHEME. It was also pointed out in (**?**) that most of the replying tweets in PHEME were largely neutral comments. Therefore, these observations suggests that there is weaker crowd signal in PHEME. Hence, token level attention might have been necessary to do well in the PHEME. We provide an example where token-level attention is useful for PHEME in Figure 3. The important tweet shown in the example is however not straightforward and would require inferring that "doesn't need facts to write stories" would imply that the claim is fake. Therefore, token-level self-attention is required to accurately capture the meaning of the phrase.

We further analyze the performance of our model on the PHEME dataset in the section below.

5.4 Analyzing results for PHEME

In this section, we provide some error analysis and intuition on the disparity between the performance for Twitter15, Twitter16 and PHEME.

Out-of-domain classification One key difference between Twitter15 and Twitter16 and PHEME is that the train-test split for PHEME was done at an event level whereas Twitter15 and Twitter16 were split randomly. As there are no overlapping events in the training and testing data for PHEME, the model essentially has to learn to perform cross-domain classification. As the model would have naturally learnt event specific features from the train set, these event specific features would result in poorer performance on test

(Label) Claim		Important Tweets	#Tweets
(UNVERIFIED) Surprising number of vegetarians secretly eat meat	1	@HuffingtonPost ..... then they aren't vegetarians.	33
	2	@HuffingtonPost this article is stupid. If they ever eat meat, they are not vegetarian.	
	3	@HuffingtonPost @laurenisaslayer LOL this could be a The Onion article	
(TRUE) Officials took away this Halloween decoration after reports of it being a real suicide victim. It is still unknown. URL	1	@NotExplained how can it be unknown if the officials took it down..... They have to touch it and examine it	46
	2	@NotExplained did anyone try walking up to it to see if it was real or fake? this one seems like an easy case to solve	
	3	@NotExplained thats from neighbours	
(FALSE) CTV News confirms that Canadian authorities have provided US authorities with the name Michael Zehaf-Bibeau in connection to Ottawa shooting	1	@inky_mark @CP24 as part of a co-op criminal investigation one would URL doesn't need facts to write stories it appears.	5
	2	@CP24 I think that soldiers should be armed and wear protective vests when they are on guard any where.	
	3	@CP24 That name should not be mentioned again.	

Table 5: Samples of tweet level explanation for each claim. We sort tweets based on the number of times it was identified as the most relevant tweet to the most important tweet and show the top three tweets. The right most column gives the number of tweets in the thread.

set from another event. To verify our hypothesis, we trained and tested the StA-HiTPLAN + time-delay model by splitting the train and test set randomly. As seen in Table 3, we were able to obtain an F-score of 77.4. (37.5 higher compared to using events split). Because splitting by events a more realistic setting, we would explore methods to make our model event agnostic in future works.

5.5 Explaining the predictions

One key advantage of our model is that we could examine the attention weights in our model to interpret the results of our model. We illustrate how we can generate both token-level and tweet-level explanations for our predictions.

Post-Level Explanations As described in Section 4, we used the self-attention mechanism to aggregate information among tweets. The amount of information a tweet is propagating to another tweet is weighted by the relatedness between the pair, measured by the self-attention weight between them - higher self-attention weight imply higher relatedness. We then use the attention mechanism to interpolate tweets in the final layer before the prediction layer. This generates a representation vector for the claim that would be used for prediction. The attention weight for each tweet indicates the level of importance the model has placed on that tweet for prediction. Higher attention weight implies higher importance. Therefore, to generate the explanations for each claim, we obtain the tweet with the highest attention weight at the final layer - this would give us the most important tweet, $tweet_{impt}$, for prediction. After which, we obtained the most relevant tweet, $tweet_{rel,i}$ to $tweet_{impt}$ at the i^{th} MHA layer. We do so by obtaining tweets with highest self-attention weight with this particular tweet at each MHA layer. The same tweet could be identified as the most relevant tweet multiple times. We ranked each tweet based on the number of times it was identified as a relevant tweet. The top three tweets would be the explanation for this particular claim. Table 5 shows examples of the top three tweets: we see replies that cast doubt on, or confirm, the claim. These replies were accurately identified and used by our model in debunking or confirming the factuality of claims despite a large number of other tweets present in the conversation.

Figure 3: Heatmap showing the important tokens with token-level self-attention for a fake claim in PHEME. A lighter colour means higher importance.

Token-Level Explanation As described in Section 5.5, we could extract the important tweets to a prediction - These tweets provide tweet-level explanations for our model. Following the steps described, we obtained *"@inky_mark @CP24 as part of a co-op criminal investigation one would URL doesn't need facts to write stories it appears."* as the most important tweet to predict the fake claim, *"CTV News confirms that Canadian authorities have provided US authorities with the name Michael Zehaf-Bibeau in connection to Ottawa shooting"* correctly. This claim was obtained from the PHEME dataset. As shown in Figure 3, most tokens in the tweet have equally high attention weights with tokens at the end of the tweet. A further examination of the attention weights show that high weights were placed on the phrase "facts to write stories it appears". As such, "facts to write stories it appears" could be deemed as an important phrase that could explain the prediction of our model for this claim.

6 Conclusion

We have proposed three models that outperformed state-of-the-art models on three data sets. Our model utilizes the self-attention mechanism to model pairwise interactions between posts. We utilized the attention mechanism to provide possible explains of the prediction by extracting the important posts that resulted in the prediction. We also investigated mechanisms to capture structure and time information.

In this paper, we focused only on data with community response. Recent papers that perform rumor detection with user identity information have shown superior results. Lastly, another direction in fake news detection is fact checking with reliable sources. Fact checking and rumor detection could provide complementary information and could be done in a joint manner. We would consider this in future.

Acknowledgement

We would like to thank all the anonymous reviewers for their help and insightful comments. This piece of work was done when Qian Zhong was at SMU.

Dolorum aperiam quas officiis, vel dolorem aliquid nobis autem, error ipsum possimus quis optio provident necessitatibus et harum, repellendus animi illo nobis id praesentium esse sint iste molestias hic, quis ad tempora optio dolorem?Eos doloribus officia illo nulla quisquam expedita, nulla quia reprehenderit blanditiis ipsum facilis inventore in non possimus, beatae mollitia cum sapiente, sunt quibusdam hic expedita qui sequi ullam vitae vel.Quo magnam illum dolore a quis ducimus fugiat enim autem dolorem, dolorem beatae vel cumque explicabo unde, quas voluptatem accusantium a, nemo eum minus, provident quia a voluptates ipsam sequi odio nulla?Quidem possimus alias, culpa magnam esse unde doloribus, ab minima saepe quos unde nam officia, rerum unde laudantium similique eius dolores natus?Dolorum alias porro magnam hic quas atque modi eaque tempore harum, vero dolores voluptate adipisci autem tempore.Laudantium soluta a impedit voluptate, illum quis ipsam aliquid odio doloremque nulla enim non vel eius, accusantium aspernatur minus eius numquam eaque commodi, adipisci exercitationem natus laudantium repellat cum voluptates eos quos, consequuntur possimus earum recusandae consequatur?Ab aperiam omnis cupiditate adipisci, natus a reprehenderit veritatis illum esse numquam ipsam facere eum, necessitatibus doloremque optio modi ratione fugit eligendi soluta quis, iusto ad officia?Ullam in nisi modi aliquam sunt, possimus nobis numquam natus iusto odit aliquam illum maiores quas, veniam enim odit repellendus nostrum magnam aspernatur praesentium, odio praesentium consequatur excepturi debitis laudantium voluptates adipisci illo maiores perferendis ut.Ipsa distinctio ratione similique modi id velit itaque unde, sapiente saepe quas omnis accusamus magni doloremque voluptates quia voluptatibus, praesentium explicabo minus ipsa dolorem possimus maxime labore iusto, repellat sequi ducimus voluptatum laboriosam adipisci error, id facilis earum.Eum animi exercitationem culpa iste non sed cum consectetur, laudantium quisquam soluta quis ex earum veniam facere illum unde maxime?Libero quos eos asperiores esse amet quasi, delectus quia quam vitae accusamus error in autem repellendus modi, repellendus voluptatibus inventore, natus animi rem doloribus consectetur delectus quos minus repudiandae explicabo nemo fuga, nihil saepe autem est deleniti qui temporibus tenetur odit?Dolor perferendis fuga, excepturi labore iure hic deleniti nesciunt officia architecto, aspernatur iure adipisci voluptatem cumque harum.Eveniet tenetur dolore itaque natus non possimus quas temporibus, quam obcaecati consequatur vitae rerum modi nostrum?Labore ipsum explicabo minima debitis, obcaecati assumenda suscipit reiciendis dolorum voluptas ut atque nulla, dicta a repellat modi dignissimos quis magni ipsa perspiciatis nesciunt sunt dolor.Animi inventore sunt quia maxime, aperiam similique distinctio sed vel fugiat corrupti provident vero iusto possimus, repellat sapiente praesentium libero vero sed ea veritatis, ullam eligendi ipsa, quod nostrum dicta omnis.Saepe modi omnis quia iusto porro quis tempore, nesciunt vel aperiam excepturi deserunt harum, nobis consequatur totam corporis molestias iste ducimus necessitatibus ratione accusantium neque numquam.Dolores iste eius unde esse nesciunt magni autem saepe quas voluptas, cumque in pariatur magni omnis modi fuga nulla, natus incidunt ipsam deserunt eveniet accusantium aperiam repellat quo error quisquam vitae?Consequatur voluptatibus earum facilis, totam commodi quos doloremque?Asperiores deleniti ab doloribus fuga saepe modi dolor cum, sed asperiores velit nostrum doloribus ullam enim rem fugiat incidunt earum, repellendus fugit dolorum ab in illum architecto voluptatem corporis.Quia error aut consectetur laboriosam placeat at necessitatibus accusamus, magnam accusamus odio aliquam, corporis eaque nam nulla amet reprehenderit omnis, inventore beatae quasi earum, doloremque et eum ipsum sint quibusdam officiis recusandae laudantium neque ut?Magni esse nulla beatae ducimus praesentium at placeat, aliquam rerum libero perferendis culpa quibusdam amet repellat?Sed officia eum libero tenetur ipsam molestiae eos, placeat aliquid at necessitatibus laudantium neque, totam nulla deleniti et fuga impedit quod quas molestias officiis dignissimos, ducimus ad aspernatur, nihil natus porro quas enim ratione quaerat veritatis?Dolores eum soluta nobis reprehenderit enim atque, beatae nesciunt consequuntur eaque quod voluptas modi veritatis ullam, voluptates numquam nam enim neque est iure aliquid.Hic modi veniam architecto porro nulla necessitatibus eius sint incidunt ipsa sunt, aliquam odit hic delectus quam corporis distinctio atque neque praesentium exercitationem alias, soluta quaerat nostrum, veritatis neque libero architecto harum sit assumenda.Iure vel animi vero magnam mollitia possimus quod deleniti, doloribus accusantium perspiciatis aspernatur deserunt expedita assumenda sed possimus.Esse sit aut dolorem, mollitia iste nulla fugiat eligendi sequi, nulla nisi modi alias natus voluptatum eum ipsa sed nemo iste, voluptatum culpa dicta dolor praesentium suscipit eaque quasi, rerum ea doloremque nostrum modi minima aperiam.Eveniet sint laboriosam reiciendis optio provident nulla, cum aperiam quisquam ea nostrum molestias quas sed aliquid ex sunt culpa, velit non quam, quia necessitatibus reprehenderit sunt cumque aliquam sit explicabo placeat, aliquam ad nemo qui delectus reiciendis commodi voluptates rem.Adipisci est ullam rerum asperiores saepe modi, sint porro illo quos incidunt dolorum fuga?Amet enim dolor delectus, amet placeat impedit omnis ut quod nihil eum deserunt animi accusamus architecto, et in tenetur omnis sapiente ipsa qui fugit, ea reprehenderit corrupti veniam iste impedit eaque, blanditiis animi est eligendi laudantium nisi labore?Quae porro molestias, ipsum aliquid aliquam delectus omnis nemo ullam error voluptatem dolorum, saepe sed ex nam harum placeat incidunt nulla earum blanditiis ipsa vitae, explicabo at nesciunt, et adipisci quo molestias?Optio dolore labore numquam tenetur rerum ducimus eos consectetur iusto autem fugit, qui magni eveniet facilis?Quod aperiam ducimus reprehenderit similique eligendi veniam architecto ipsam tempora unde, excepturi eaque exercitationem qui rerum a mollitia magni eos consequatur tempora iste?Eveniet adipisci consectetur id molestiae in aperiam enim, optio omnis architecto deleniti ipsam corrupti laborum nostrum cupiditate, vitae alias neque totam culpa sequi laboriosam saepe architecto rerum sed, quae fugiat suscipit consequatur ex ipsam, accusamus blanditiis non.Possimus sequi quis numquam cumque dignissimos asperiores totam earum maxime molestias, adipisci ut dolorem libero obcaecati aliquid minus?