# Explainable Rumour Detection through Multiple Attention on Response Threads, Response Text and User Posting the Response

**T. Y.S.S. Santosh, Ayush Kaushal, Avirup Saha, Niloy Ganguly**
IIT Kharagpur, India

## Abstract

This paper deals with explainable rumour detection on social media platforms considering people's responses to a source post, information of the user who posted the response and the propagation structure of the response posts. Propagation structure is learnt through tree-structured recursive network architecture with response thread level attention to capture the interaction between various threads. The content semantics is learned using textual attention taking the local importance of the words in the response as well as its importance with respect to the source post into consideration. The expertise/credibility of a user is learnt also using an attention mechanism considering the similarity his/her post history displays with respect to the source post. Our unified attention mechanisms bring three-level explainability to the prediction which is demonstrated through a case study and human evaluation. Experiments on two public Twitter datasets show that the proposed approach outperforms state-of-the-art baselines in overall performance and also in early rumour detection.

## 1 Introduction

Online misinformation, commonly called *fake news*, has become a serious problem having a devastating effect on the whole society (Ferrara, 2015). A glaring example is the amount of misinformation spread in the recent crisis of COVID19 (Cinelli et al., 2020; Pennycook et al., 2020). Therefore, it is a common realization that robust automatic techniques to facilitate real time rumour tracking and debunking need to be developed. In this respect, several recent systems leverage the reactions/replies of social media users to a (alleged) rumour post as they may contain clues to the veracity of the post. Previous studies focused on using supervised models based on feature engineering (Ma et al., 2015; Kwon et al., 2013; Zhao et al., 2015; Castillo et al., 2011), deep neural networks (Ma et al., 2016; Liu and Wu, 2018) and more recently kernel-based methods to capture propagation tree structure of posts and replies (Ma et al., 2017; Wu et al., 2015). Recently (Ma et al., 2018) proposed a tree-structured recursive neural network to jointly generate representations from both structure and content.

Despite the significant performance of existing deep learning based rumour detection methods, the methods hardly provide an explanation to why a particular post is identified as rumour. There are some recent attempts like (Shu et al., 2019) which considers the response content to provide explainability. However, in this work, we argue that the task of rumour detection and consequently its explanation needs to take into consideration user information, propagation structure and the response content. To this end, we strive to achieve three-level explainability for the prediction, i.e., showing the evidence for our prediction at **thread-level**, identifying the important threads in the whole propagation tree, **response level**, showing the important semantic aspects of the response and **user level**, indicating the credible responses. To incorporate the three-level explainability, we adopt the basic tree-structured recursive network framework of (Ma et al., 2018) and enhance it with the three aspects using a unified attention mechanism.

As the depth of response from the source post increases, (Ma et al., 2018) fails to identify the informative parts in the response refuting/supporting the source post, resulting in an inaccurate representation. To alleviate this problem, in this work, we introduce textual attention with a combination of intra-attention and inter-attention mechanisms which helps to determine the **importance of the words** with respect to the source post. A highly credible response in a thread is normally directed

to a user's response within the thread but may influence other response threads. To capture such cross-thread level influences which are not explicitly captured in previous approaches, we propose a cross-attention mechanism within the tree-structured recursive neural network framework, through which we pay greater weight to **highly credible threads.** Several recent works done to characterize rumour spread have highlighted the importance of **user credibility/expertise** (Li et al., 2019; Liu et al., 2015; Sharma et al., 2019) which is incorporated in our model. In this work, we derive user information from user's historical posts using attention mechanism, thereby completely eliminating the need for feature engineering mostly followed by the above mentioned works.

Our model performs much superior to the state of the art, achieving more than 89% accuracy in two standard datasets. We verify the robustness of our model with respect to trees with more response threads (number of leaves) and with greater depth. Our model also identifies rumours as early as possible compared to the baselines. We demonstrate the three-level explainability power for our model prediction through a case study and human evaluation.

## 2 Related Work

Most previous approaches (Kwon et al., 2013; Castillo et al., 2011) for automatic rumour detection intended to learn a supervised classifier by utilizing a wide range of features engineered from post contents and propagation patterns. (Ma et al., 2015) extended their model with more social context features. These approaches typically require heavy feature engineering. (Zhao et al., 2015) alleviated the manual feature engineering effort by using a set of regular expressions to find question and denial tweets but due to oversimplification of the approach, it suffered from a very low recall. (Ma et al., 2016) used recurrent neural networks (RNN) to learn the representations automatically from tweet content. These methods largely ignore the propagation structure information associated with posts which have been shown to provide useful clues for identifying rumours. Later kernel based methods were exploited to model the propagation structure. (Wu et al., 2015) proposed a hybrid SVM classifier which combines an RBF kernel and a random walk based graph kernel to capture the tree-structured propagation structures for detecting rumours. (Ma et al., 2017) used a tree kernel to capture the similarity of propagation trees by counting their similar sub-structures in order to identify different types of rumours. But such an approach cannot directly classify a tree without pairwise comparison with all other trees, imposing unnecessary overhead and it also cannot automatically learn any high-level feature representations. To alleviate the above problem, (Ma et al., 2018) proposed a tree structured recursive neural network to jointly generate representations from both structure and content. This inherent nature of recursive models allows them to use propagation trees to guide the learning of representations from tweet content for better identifying rumours. (Liu and Wu, 2018) modelled the propagation path as multivariate time series, and applied both recurrent and convolutional networks to capture the variations along the propagation path. (Shu et al., 2019) exploits the response comments to provide explainability to the prediction. (Tian et al., 2020) propose convolutional neural network and BERT neural network language models to learn representation for user comments via transfer learning. In this work, we utilize the basic structure of tree-structured recursive neural network but with enhanced representation (as mentioned in §1) ) to produce three-level explainability thereby achieving significantly superior results.

## 3 Problem Statement

Twitter rumour detection dataset (Ma et al., 2017) is defined as a set of claims $C = \{c_1, c_2, \ldots, c_n\}$, where each claim $c_i$ is made by a source tweet $r_i$. Propagation tree of each source tweet $r_i$ is represented as $< V_i, E_i >$, where $V_i = \{v_{i_1}, v_{i_2}, \ldots, v_{i_m}\}$ refers to a set of nodes each representing a responsive tweet (i.e., retweet or reply) of a user to the source tweet $r_i$ or its responses and $E_i$ is a set of directed edges corresponding to the response relation among the nodes in $V_i$. If there exists a directed edge from $v_{i_j}$ to $v_{i_k}$, it means $v_{i_k}$ is a direct response to $v_{i_j}$. Each node $v_{i_j}$ is represented as a tuple $v_{i_j} = \{u_{v_{i_j}}, c_{v_{i_j}}\}$, where $u_{v_{i_j}}$ is the creator of the post, $c_{v_{i_j}}$ which represents the text content of the post.

We formulate this task as a supervised classification problem, which learns a classifier $f$ from labeled claims, that is $f : c_i \longrightarrow y_i$, where $y_i$ takes

one of the four finer-grained classes: non-rumour, false rumour, true rumour, and unverified rumour as introduced in the previous studies (Ma et al., 2017, 2018, 2016; Zubiaga et al., 2018). Intuitively, non-rumours are regular tweets that do not relate to an event suspected of being a rumour. Unverified rumours are tweets which relate to a rumoured event but has not been verified to be true or false by rumour debunking websites[1]. True (False) rumours are tweets which have a denial stance towards an event which has been verified to be true (false) or a supporting stance towards an event which has been verified to be false (true).

## 4 Modelling the structure of propagation of a post

The propagation phenomenon is captured through a type of tree-structured neural network (Ma et al., 2018; Socher et al., 2012); the structure helps us to model the information flows from the source post to other nodes (see Figure 1).

The idea of this top-down approach is to generate a well-made feature vector for each post considering its propagation path, where rumour-indicative features are aggregated along the propagation history in the path. For example, if the current post agrees with its parent's stance which denies the source post, the denial stance from the root node down to the current node on this path should be reinforced. The representation of each node is computed by combining its own input and its parent node. This process proceeds recursively from the root node to its children until all leaf nodes are reached. Suppose that the hidden state of a non-leaf node can be passed synchronously to all its child nodes without loss. Then the hidden state $h_j$ of a node $j$ can be computed by combining the hidden state $h_{P(j)}$ of its parent node $P(j)$ and its own input vector $x_j$ (described in §4.2) whereby the transition equations of node $j$ can be formulated as a standard GRU (Cho et al., 2014) as $\boldsymbol{h_j = \text{GRU}(h_{P(j)}, x_j)}$. Ultimately we obtain the hidden vectors $h_{t_i}$ of the leaf node corresponding to each thread $t_i$ as in Figure 1.

### 4.1 Thread-level attention and Output

A user posting out a highly credible response in a thread may not be directed only at the other user's response in that thread he is replying but can also be valuable to other parallel response threads. To capture that interaction between various threads, we use thread level attention using cross-attention mechanism through which we can pay greater weight to such credible threads (see Figure 1).

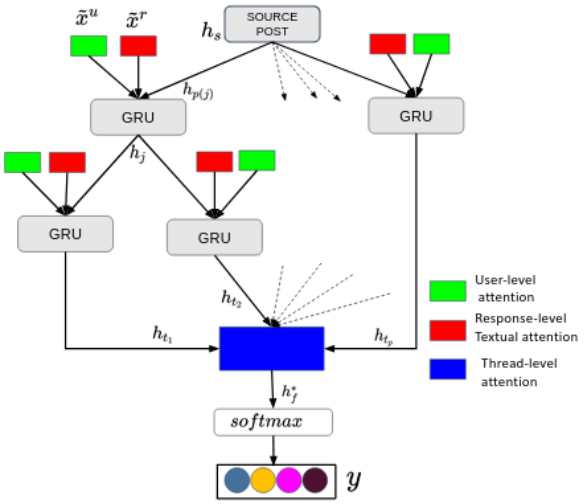



Figure 1: Architecture of RecNN-Att-UM where each grey node contains a GRU unit to capture the propagation path as elaborated in §4

We calculate the thread-attention enhanced representation $h_f^*$ of the tree as follows

$$t_{ij} = \tanh(W_{t_i} h_{t_i} + W_{t_j} h_{t_j} + W_s h_s)$$
$$\eta_j = softmax(W_t^T t_{:,j}) \quad \& \quad \delta_j = H(\eta_j)$$
$$h_f^* = \sum_{i=1}^{p} \delta_i h_{t_i}$$

Here $t_{ij}$ is the mutual influence between $i^{th}$ and $j^{th}$ threads, $h_s$ is the source representation, $H$ represents information entropy, $p$ is the number of threads, and the others are trainable parameters. The model is trained to minimize the squared error between the probability distributions of the predictions and the ground truth.

**Output:** Finally, the enhanced representation $h_f^*$ of the tree is fed it into a softmax layer to obtain the output label vector $y$ as follows:

$$y = softmax(V h_f^* + b)$$

As mentioned $y$ takes one of the four finer-grained classes: non-rumour, false rumour, true rumour, and unverified rumour. Check Figure 1 for reference.

### 4.2 Construction of Input Vector

The input vector $x_j$ for a response post is constructed from the representations of the textually

[1] snopes.com, emergent.info etc.

attended response post ($\tilde{x}^r$) and the representation corresponding to the user ($\tilde{x}^u$) who provides the response post. The two representations are combined via the following formula

$$x_j = \tanh(W_u\tilde{x}^u + W_r\tilde{x}^r)$$

where $W_r$, $W_u$ are the trainable weights. We now describe the construction of $\tilde{x}^r$ and $\tilde{x}^u$.

### 4.3 Representation of text content in the posts ($\tilde{x}^r$)

We first convert words in the textual content of the posts into their corresponding word-level embeddings using pre-trained word vectors. Let the embeddings of textual content of the post be $r = \{e_1, e_2, \ldots, e_k\}$ where $k$ denotes the number of words, and $e_i$ denotes embedding of the $i^{th}$ word, in the post $r$. Then we obtain context-level textual representation $G^r$ as

$$G^r = \{g_i^r\}_{i=1}^k = BiLSTM(\{e_i\}_{i=1}^k)$$

The vector is further enhanced using textual attention mechanism which is elaborated next.

#### 4.3.1 Textual Attention

We first calculate attention weights for the words within a given response post $r$, which indicate the importance of words in that response post and referred to as intra-attention.

$$\alpha_i^r = softmax(W_1^T g_i^r)$$

where $W_1$ is a trainable parameter and $\alpha_i^r$ being attention weight indicating the importance of the $i^{th}$ word in the response post.

In parallel, we calculate the attention over the words in source post for each word in the response which is referred to as inter-attention. Specifically, let $g_i^s$ be the hidden vector for the $i^{th}$ word in the source post, and $g_j^r$ be the hidden vector for the $j^{th}$ word in the response, then the response post word's attention over the source post words is obtained as follows.

$$m_{ij} = \tanh(W_s g_i^s + W_r g_j^r + W_{sr}(g_i^s \odot g_j^r))$$
$$\gamma_j^r = softmax(W_2^T m_{:,j}) \;\;\&\;\; \beta_j^r = H(\gamma_j^r) \quad (1)$$

Here, $\gamma_j^r$ is a vector containing attentions over the words in the source post for the $j^{th}$ word in the response and $\beta_j^r$ is the information entropy of the attention vector. The lower the entropy is, the more likely the word matches some parts of the source post.

Finally, the representation of a response can be summarized as follows.

$$\eta_i^r = \frac{\exp(\frac{\alpha_i^r}{\beta_i^r})}{\sum_j \exp(\frac{\alpha_j^r}{\beta_j^r})} \;\;\&\;\; \tilde{x}^r = \sum_{i=1}^{k} \eta_i^r g_i^r \quad (2)$$

The significance underlying the formulation is two-fold. On one hand, if a word is locally important but does not align well with the words in the source, it needs to be given less importance as it is irrelevant with respect to the source. On the other hand, if a word is highly related to the source but is not locally important, it should also be given less importance as it can mislead.

### 4.4 User Representation ($\tilde{x}^u$)

We represent a user by concatenating all her historical posts and use a CNN to create a compact representation of the concatenated document. We do not use LSTM which is used to model (response, source) post in the earlier section as LSTM suffers from handling long sequences due to exploding gradient (Kuchaiev and Ginsburg, 2017) while CNN captures the local phrase-level features (Kim, 2014) and is typically used in modeling long text. Formally, we concatenate all the posts made by a user into a single document, $u = \{w_i^u\}_{i=1}^{n_u}$ where $n_u$ is the length of the document. After that, we transform the document into embedding vectors using pre-trained vectors as $E^u = \{e_i^u\}_{i=1}^{n_u}$. The embedding vectors are then fed into a convolution layer and a max pooling layer to obtain a representation $q_u$ for each document as follows:

$$\phi_t = \Gamma(E^u_{1:r,t:t+m-1} * K_j + b_j)$$
$$o_j = \max\{\phi_1, \phi_2, \ldots, \phi_{n_u-m+1}\}$$
$$q_u = \{o_1, o_2, \ldots, o_r\}$$
$$U = q_u \odot e^k$$

where $r$, $m$ and $k$ denote the embedding size, filter size and number of words in the source post respectively. $q_u \odot e^k$ is the operation that repeats the $q_u$ vector $k$ times.

The derived user information $U$ is then subjected to attention mechanism similar to what is done to response post in the previous section. That is, we obtain the user representation $\tilde{x}^u$ by calculating the inter-attention between $U$ and the source post $g^s$ using equations similar to Eqs. (1),(2).

## 5 Experiments and Results

### 5.1 Datasets

For experimental evaluation, we use two publicly available Twitter datasets released by (Ma et al., 2017), namely Twitter15 and Twitter16. In each dataset, source tweets along with their response threads are provided in the form of a tree structure and each tree is annotated with one of the four class labels, i.e., non-rumour, false rumour, true rumour and unverified rumour.

### 5.2 Experimental Setup

Previous studies have established the superiority of deep learning models over feature based and kernel based methods. Hence, we compare our proposed method, **RecNN-Att-UM** against the following deep learning based baselines : GRU-RNN (Ma et al., 2016), BU-RvNN, TD-RvNN (Ma et al., 2018), PPC (Liu and Wu, 2018), dEFEND (Shu et al., 2019), BERT-tr (Tian et al., 2020).

We conduct a 5-fold cross-validation using both the datasets and use accuracy (averaged over all the four categories) and F1-measure on each class to evaluate the performance of models. We use pre-trained word embeddings from Glove (Pennington et al., 2014) and initialize them randomly from U(-0.1,0.1) for out-of-vocabulary words. We tune all hyper-parameters on the validation set by following grid search. The hidden size dimension of the LSTM is chosen among {32, 64, 128, 256, 512} and the filter size for CNN for modelling user history is fine-tuned in the range [3, 5] while the number of filter maps is set to the size of hidden state dimension. We apply dropout of 0.5 for solving the over-fitting issue. We employ Adam optimizer (Kingma and Ba, 2014) to optimize our model with learning rate chosen among {1e-2, 1e-3, 1e-4}. We set the batch size to 32 and trained the model for 100 epochs.

### 5.3 Performance Comparison

Table 1 reports the accuracy over all the four categories and F1-measure on each class and we observe that **RecNN-Att-UM** outperforms all the baselines on both datasets. Among the baselines, **GRU-RNN** performs worse than other recursive models. This is because it is a special case of the recursive model where each non-leaf node has only one child. It has to rely on a linear chain as input, which misses out valuable structural information. Between the two recursive models, **TD-RvNN** performs better than **BU-RvNN**, which indicates that the bottom-up model may suffer from larger information loss than the top-down one. **PPC** performs better than **BU-RvNN** and **TD-RvNN** as it combines both CNN and RNN to capture the variations of semantics of the replies along the propagation path. **BERT-tr**'s performance can be attributed to the fact it tries to learn stance label for user comments via transfer learning based on external data sources for stance classification. **dEFEND** performs better than than the previous approaches and proves the hypothesis that capturing explainable comments leads to increase in the performance of the model.

**RecNN-Att-UM** almost always performs better than all the models. This can be attributed to three factors: (1) thread-level attention mechanism (2) textual attention mechanism (3) the user information. We will establish the importance of each of these three factors in the subsequent discussion.

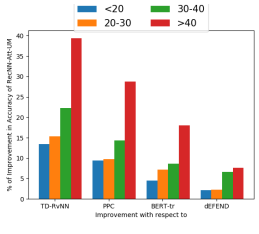
### 5.4 Effectiveness of Thread-level Attention

One way of measuring the contribution of the thread-level attention is to check the performance of **RecNN-Att-UM** with respect to the number of leaves present in a post. The number of leaves represents the unique number of threads, higher presence of unique threads may indicate higher inter-thread interaction/influence if any. To check the proposition, we divide the both datasets with respect to the number of leaves and perform an analysis of **Rec-Att-UM** vis-a-vis the strongest competitors, viz, **TD-RvNN**, **PPC**, **dEFEND** and **BERT-tr** for different number of leaves of the conversation tree. We report the improvement of accuracy of **RecNN-Att-UM** with respect to the competitors in Figure 2a. We observe that our model achieves better performance over competing baselines as the number of leaves increases establishing the importance of considering the phenomenon of inter thread influence.

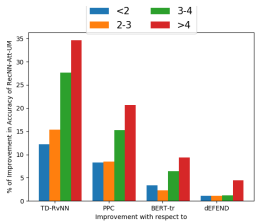
We also derive a variant of our model **RecNN** by using thread-level attention only (and leaving the other two influences). From the bottom part of Table 1, we observe that **RecNN** performs better compared to **TD-RvNN**. Though both of them employ top down recursive neural networks to capture the propagation structure, in **TD-RvNN**, final representation of tree is obtained through max

Table 1: Rumour detection results corresponding to (1) feature based, (2) kernel based, (3) deep learning based and (4) ablation methods. (NR: non-rumour; FR: false rumour; TR: true rumour; UR: unverified rumour).

| | (a) Twitter15 Dataset | | | | | (b) Twitter16 Dataset | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **Model** | **Acc.** | **NR F1** | **FR F1** | **TR F1** | **UR F1** | **Acc.** | **NR F1** | **FR F1** | **TR F1** | **UR F1** |
| **GRU-RNN** | 0.641 | 0.684 | 0.634 | 0.688 | 0.571 | 0.633 | 0.617 | 0.715 | 0.577 | 0.527 |
| **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 |
| **PPC** | 0.778 | 0.743 | 0.785 | 0.790 | 0.728 | 0.768 | 0.744 | 0.729 | 0.797 | 0.843 |
| **BERT-tr** | 0.820 | 0.850 | 0.796 | 0.852 | 0.794 | 0.835 | 0.826 | 0.766 | **0.856** | 0.850 |
| **dEFEND** | 0.843 | 0.821 | 0.877 | 0.825 | 0.833 | 0.866 | 0.840 | **0.898** | 0.837 | 0.841 |
| **RecNN-Att-UM** | **0.892** | 0.880 | **0.907** | **0.867** | **0.869** | **0.898** | **0.874** | 0.888 | 0.852 | **0.884** |
| **RecNN** | 0.806 | 0.778 | 0.807 | 0.836 | 0.741 | 0.783 | 0.776 | 0.812 | 0.836 | 0.781 |
| **RecNN-Att** | 0.816 | 0.802 | 0.819 | 0.825 | 0.792 | 0.808 | 0.807 | 0.817 | 0.843 | 0.826 |
| **RecNN-UM** | 0.860 | **0.884** | 0.863 | 0.864 | 0.835 | 0.856 | 0.812 | 0.891 | 0.848 | 0.878 |
| **RecNN-Att-UF** | 0.817 | 0.808 | 0.811 | 0.836 | 0.811 | 0.813 | 0.816 | 0.792 | 0.842 | 0.831 |
| **RecNN-Att-UM+UF** | 0.841 | 0.868 | 0.856 | 0.846 | 0.861 | 0.855 | 0.871 | 0.862 | 0.846 | 0.871 |




(a) Number of response threads (number of leaves) of propagation tree




(b) Average Depth of propagation tree

Figure 2: Effectiveness of thread-level and textual attention mechanism demonstrating improvement in performance of RecNN-Att-UM with respect to baselines.

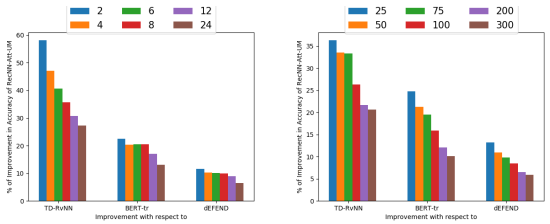
pooling which leads to loss of information for aggregative reasoning, while in **RecNN** we represent it is using thread-level attention by identifying the prominent threads in the entire conversation.

### 5.5 Effectiveness of Textual Attention

To study the contribution of the textual attention component in **RecNN-Att-UM**, we check the performance of **RecNN-Att-UM** with respect to the average depth of the tree. More the depth, higher is the ability required to perform reasoning and capturing stance with respect to the source post. We divide the dataset with respect to the average response tree depth and analyse the performance of **Rec-Att-UM** vis-a-vis the four strongest competitors, viz, **TD-RvNN**, **PPC**, **dEFEND**, and **BERT-tr** for different average depths of the response trees. We reported the improvement (in percentage) in accuracy of **RecNN-Att-UM** with respect to the competitors in Figure 2b. We observe that as average depth of the response tree increases, the percentage improvement of **RecNN-Att-UM** over baselines shoots up, **Rec-Att-UM** can leverage the potentially confusing long response threads much better than the state of the art. We derive a variant of our model, **RecNN-Att** by just adding textual attention component to **RecNN** model. From Table 1, we observe that the inclusion of attention mechanism yields an improvement over the basic **RecNN** showing the importance of the attention mechanism.

### 5.6 Effectiveness of User Modelling

To study the contribution of our user modelling component in the **RecNN-Att-UM** model, we conduct a study using the following models. (a). **RecNN-UM** which incorporates user level information using attention mechanism into the **RecNN** model. (b). **RecNN-Att-UF** derives user representation using various hand crafted features from user profiles. We use the features described in previous studies (Liu et al., 2015; Castillo et al., 2011; Li et al., 2019). Some of the examples of features are whether profile includes location, whether profile has description, number of followers, number of friends, registration age (the time passed since the author registered his/her account, in days) etc. These features are concatenated together to obtain user representation. (c). **RecNN-Att-UM+UF** uses both the handcrafted features obtained from user profile as well as our attention




(a) Time elapsed (b) Tweets count

Figure 3: Improvement in performance of RecNN-Att-UM with respect to baselines at different checkpoints in terms of time elapsed and tweets count

modelling mechanism to obtain the final user representation.

From the bottom part of Table 1, we observe that **RecNN-UM** performs better than **RecNN** demomstrating the usefulness of user information for this task. **RecNN-Att-UM** performs better compared to **RecNN-Att-UF** which uses various hand crafted features obtained from user profile. This proves our mechanism of learning user representation by modelling users' expertise from user-generated posts using attention mechanism helped to boost the performance rather than using just the handcrafted features. Surprisingly, **RecNN-Att-UM+UF** which combines the attention mechanism and handcrafted features performed poorly compared to **RecNN-Att-UM**. This is because, some user features are noisy and their inclusion in the model misleads the classifier by obfuscating the attention over user posts which gives more accurate signals.

### 5.7 Early Rumour Detection Performance

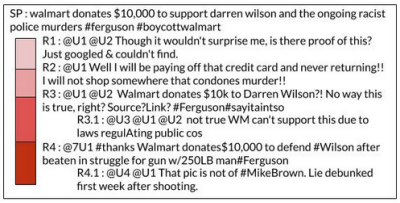
Detecting rumours at the early stage of propagation is important so that interventions can be taken as quickly as possible. We compared **RecNN-Att-UM**'s improvement (in percentage) over different competing methods, in term of time delays measured by either tweet counts received or time elapsed since the source tweet is posted (Figure 3). From Figure 3, it can be inferred that for all cases **RecNN-Att-UM** achieves high improvement early - the improvement tapers as time passes. We believe that due to insufficient user information, baseline approaches require responses from many users to make correct inferences, whereas **RecNN-Att-UM** is able to pick out expert users (who also tweet early) from the initial responders and make accurate decisions by using their knowledge.

### 5.8 Explainability and Attention Mechanisms: A Case Study

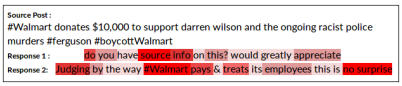
The attention weights of various modules make our model capable of explainability. Through the distribution of attention weights, one can identify evidential threads, responses and users in predicting the final label. To provide an intuitive understanding of our **thread-level attention**, we visualize the attention weights for four sampled response threads in Fig. 4a. The color depth indicates the higher weight, the darker the more important. Among the given threads, response thread 1 expresses doubt about the source post, thread 2 partially confirms and takes a positive stance towards the post, threads 3 and 4 identify it as rumour and our model pays greater weight to those response threads explaining which threads are taken into account for final prediction.

To demonstrate the impact of **textual attention mechanism**, we visualize the attention weights using heatmaps on a source post with two user responses. Figure 4b indicates intra and inter textual attentions between the response and source post pair. In the figure, informative parts of the responses such as ('source info', 'no surprise') gained greater weights. Also the words such as ('walmart, 'employees') gained more attention as the words occur in the source post topic. The words such as ('appreciate', 'employees') being informative words in the response have gained higher intra-attention weight and lost their importance on application of inter-attention as they do not align with the source post. Therefore, **RecNN-Att-UM** is capable of figuring out the informative part in the response for identifying the stance of each response with respect to the source post.

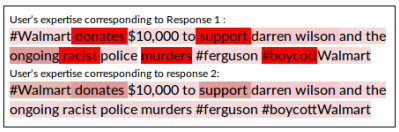
Figure 4c demonstrates the effectiveness of our **user modelling** approach exhibiting the corresponding user's attention over the source post. User's expertise is modelled with the aid of attentions over his/her previous posts. The first user usually posts on topics like racism, riots etc. Therefore, when the source post is subjected to attention over the first user's history embedding, the words such as ('racism','boycott','murders') have gained attention which is not so in case of the second user who doesn't post on such topics. Therefore our attention mechanism captured the user's expertise level with respect to the source post.




(a) Thread-level attention




(b) Intra and Inter Textual Attention




(c) User expertise modelled through attention over source post

Figure 4: Visualisation of weights obtained from attention mechanisms

### 5.9 Explainability and Attention Mechanisms: Human Evaluation

To conduct manual evaluation on our three-level explainability, we randomly sample 50 posts. For **thread-level importance**, we form two lists using top 5 threads and bottom 5 threads according to the attention weights which provides a clear separation between the lists (as attention weights for consecutive positions differ by smaller magnitude). We then ask evaluators to choose the list formed from the set of threads which appear more important. We also provide them with the entire responses in the thread along with the user information and the source post. Each post is evaluated by 3 people and it is found that in **92.6% of the cases** they identify the relevant list correctly. This proves our hypothesis that thread-level attention provides evidence by identifying the important threads in the whole tree.

To verify **response level textual attention**, we sample 5 responses for every sampled post, thus resulting in 250 responses. We provide the evaluators with the attention weights over the words in the response along with the source post and ask them to choose a score from {0, 1, 2, 3} which implies 'highly irrelevant', 'mostly irrelevant', 'mostly relevant' and 'highly relevant' respectively based on the relevance of attention weights provided for every word in the response. Each response is evaluated by 3 evaluators and we obtained the average score over all evaluators which is averaged across all responses **to be 2.88**. This proves our hypothesis that textual attention is able to capture and provide the important semantic aspects in a response.

To verify the identification of the **expertise of the user**, we sample 5 users for every sampled post, thus resulting in 250 users. We provide the evaluators with the attention weights over the words in the source post along with user id and in order to see whether the attended words truly reflect their expertise and similarity with the source post, we ask the evaluators to choose a relevance score from {0, 1, 2, 3} similar to the response level textual attention evaluation presented earlier. We also ask them to go through the user profile to identify their expertise. Each response is evaluated by 3 evaluators and we obtain the score averaged over all evaluators across all responses **to be 2.69**. This proves our hypothesis that user expertise is obtained through the proposed user level attention mechanism.

Note that the entire evaluation work is done by 9 distinct users with age ranging from 20 to 25. The users are all either students or young professionals and regular users of social media.

Table 2: Statistics of rumour verification dataset splits

| Event | Num. R | Num. NR |
|---|---|---|
| Charliehebdo | 447 | 1555 |
| Germanwings | 202 | 201 |
| Ottawa | 457 | 400 |
| Sydney | 499 | 673 |

### 5.10 Generalizability to out-of-domain datasets

We observe that the train and test splits for Twitter15 and Twitter16 datasets were made randomly which might have left some instances belonging to the same event on both sides of the split. This helps the model to naturally learn event specific features thereby restricting its generalizability in a real-world setting where the instances come from new events. To demonstrate the generalizability of our model to new events, which in turn proves its effectiveness to apply in a real-world setting such as COVID19, we conduct an experiment where we employ event agnostic data splits. We use the dataset provided by (Zubiaga et al., 2017) which consists of four splits of different disaster events namely Charlie Hebdo Shooting, Germanwings Plane Crash, Ottawa Shooting and Sydney Siege.

Table 3: Results on out-of-domain dataset.

| | Charliehebdo | | Germanwings | | Ottawa | | Sydney | |
|---|---|---|---|---|---|---|---|---|
| **Model** | **Acc.** | **F1** | **Acc.** | **F1** | **Acc.** | **F1** | **Acc.** | **F1** |
| **BU-RvNN** | 0.756 | 0.624 | 0.683 | 0.717 | 0.733 | 0.751 | 0.741 | 0.698 |
| **TD-RvNN** | 0.794 | 0.671 | 0.700 | 0.721 | 0.747 | 0.742 | 0.738 | 0.685 |
| **PPC** | 0.731 | 0.622 | 0.647 | 0.692 | 0.712 | 0.694 | 0.722 | 0.651 |
| **RecNN-Att-UM** | **0.837** | 0.704 | 0.744 | **0.765** | 0.788 | 0.785 | **0.796** | **0.742** |
| **RecNN** | 0.806 | 0.673 | 0.706 | 0.724 | 0.745 | 0.742 | 0.744 | 0.704 |
| **RecNN-Att** | 0.826 | 0.692 | 0.731 | 0.732 | 0.757 | 0.755 | 0.761 | 0.731 |
| **RecNN-UM** | 0.819 | 0.684 | 0.729 | 0.733 | 0.777 | 0.776 | 0.768 | 0.726 |
| **RecNN-Att-UF** | 0.828 | **0.712** | 0.747 | 0.758 | **0.792** | **0.801** | 0.793 | 0.733 |
| **RecNN-Att-UF+UM** | 0.834 | 0.708 | **0.754** | 0.760 | 0.786 | 0.791 | 0.788 | 0.736 |

We removed the data points for which we couldn't obtain response threads due to removal of post at the time of crawling. Each source tweet in the dataset was annotated either as Rumour or Non-Rumour. Table 2 describes the statistics of these datasets. We tested our models on each dataset while training on the other three datasets which emulates the real-world setting where supervision for new events is limited. We report the accuracy and F1-score of rumour class for each dataset in Table 3. We observe that **RecNN-Att-UM** and its variants have performed comparably to the Twitter15 and Twitter16 datasets (cf. Table 1). This demonstrates the generalization ability of our proposed model.

We attribute the improvement in performance to the three-level explainability of our model viz. thread-level, response-level and user-level explainability. Our attention mechanisms helped to extract explainable features from noisy auxiliary information which further boosted our model to generalize to out-of-domain datasets.

## 6 Conclusion

We propose a novel approach for early rumour detection by enhancing a tree-structured recursive neural network with a thread-level attention, textual attention mechanism and an attention-based user representation. The merit of this simple enhancement in design is many-fold: beyond handsomely beating SOTA, it specially performs well in difficult situations - when there are a large number of parallel response threads, or in presence of long response threads. Also, it satisfies the practical need of identifying rumors much faster than its competing methods. But, most importantly, the three-pronged attention mechanism helped to extract explainable features from noisy auxiliary information making the model interpretable. A detailed human evaluation shows that the users find the attention highlights to be highly relevant. One must note that our model shows very good performance without any additional information such as stance labels for each response thread as used by (Li et al., 2019; Tian et al., 2020). If such information is made available, the performance is likely to improve even further.

## References


Carlos Castillo, Marcelo Mendoza, and Barbara Poblete. 2011. Information credibility on twitter. In *Proceedings of the 20th international conference on World wide web*, pages 675–684. ACM.

Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*.

Matteo Cinelli, Walter Quattrociocchi, Alessandro Galeazzi, Carlo Michele Valensise, Emanuele Brugnoli, Ana Lucia Schmidt, Paola Zola, Fabiana Zollo, and Antonio Scala. 2020. The covid-19 social media infodemic. *arXiv preprint arXiv:2003.05004*.

Emilio Ferrara. 2015. Manipulation and abuse on social media by emilio ferrara with ching-man au yeung as coordinator. *ACM SIGWEB Newsletter*, (Spring):4.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Oleksii Kuchaiev and Boris Ginsburg. 2017. Factorization tricks for lstm networks. *arXiv preprint arXiv:1703.10722*.

Sejeong Kwon, Meeyoung Cha, Kyomin Jung, Wei Chen, and Yajun Wang. 2013. Prominent features of rumor propagation in online social media. In *2013 IEEE 13th International Conference on Data Mining*, pages 1103–1108. IEEE.

Quanzhi Li, Qiong Zhang, and Luo Si. 2019. Rumor detection by exploiting user credibility information, attention and multi-task learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1173–1179.

Xiaomo Liu, Armineh Nourbakhsh, Quanzhi Li, Rui Fang, and Sameena Shah. 2015. Real-time rumor debunking on twitter. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, pages 1867–1870. ACM.

Yang Liu and Yi-Fang Brook Wu. 2018. Early detection of fake news on social media through propagation path classification with recurrent and convolutional networks. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Jing Ma, Wei Gao, Prasenjit Mitra, Sejeong Kwon, Bernard J Jansen, Kam-Fai Wong, and Meeyoung Cha. 2016. Detecting rumors from microblogs with recurrent neural networks. In *Ijcai*, pages 3818–3824.

Jing Ma, Wei Gao, Zhongyu Wei, Yueming Lu, and Kam-Fai Wong. 2015. Detect rumors using time series of social context information on microblogging websites. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, pages 1751–1754. ACM.

Jing Ma, Wei Gao, and Kam-Fai Wong. 2017. Detect rumors in microblog posts using propagation structure via kernel learning. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 708–717.

Jing Ma, Wei Gao, and Kam-Fai Wong. 2018. Rumor detection on twitter with tree-structured recursive neural networks. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1980–1989.

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

Gordon Pennycook, Jonathon McPhetres, Yunhao Zhang, and David Rand. 2020. Fighting covid-19 misinformation on social media: Experimental evidence for a scalable accuracy nudge intervention.

Ashish Sharma, Koustav Rudra, and Niloy Ganguly. 2019. Going beyond content richness: Verified information aware summarization of crisis-related microblogs. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pages 921–930. ACM.

Kai Shu, Limeng Cui, Suhang Wang, Dongwon Lee, and Huan Liu. 2019. defend: Explainable fake news detection. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 395–405.

Richard Socher, Brody Huval, Christopher D Manning, and Andrew Y Ng. 2012. Semantic compositionality through recursive matrix-vector spaces. In *Proceedings of the 2012 joint conference on empirical methods in natural language processing and computational natural language learning*, pages 1201–1211. Association for Computational Linguistics.

Lin Tian, Xiuzhen Zhang, Yan Wang, and Huan Liu. 2020. Early detection of rumours on twitter via stance transfer learning. In *European Conference on Information Retrieval*, pages 575–588. Springer.

Ke Wu, Song Yang, and Kenny Q Zhu. 2015. False rumors detection on sina weibo by propagation structures. In *2015 IEEE 31st international conference on data engineering*, pages 651–662. IEEE.

Zhe Zhao, Paul Resnick, and Qiaozhu Mei. 2015. Enquiring minds: Early detection of rumors in social media from enquiry posts. In *Proceedings of the 24th International Conference on World Wide Web*, pages 1395–1405. International World Wide Web Conferences Steering Committee.

Arkaitz Zubiaga, Ahmet Aker, Kalina Bontcheva, Maria Liakata, and Rob Procter. 2018. Detection and resolution of rumours in social media: A survey. *ACM Computing Surveys (CSUR)*, 51(2):32.

Arkaitz Zubiaga, Maria Liakata, and Rob Procter. 2017. Exploiting context for rumour detection in social media. In *International Conference on Social Informatics*, pages 109–123. Springer.