An Attention-based Rumor Detection Model with Tree-structured Recursive Neural Networks

JING MA, Dept. of Computer Science, Hong Kong Baptist University, Hong Kong SAR, China
WEI GAO, School of Information Systems, Singapore Management University, Singapore
SHAFIQ JOTY, Nanyang Technological University; Salesforce Research, Singapore
KAM-FAI WONG, Dept. of SEEM, The Chinese University of Hong Kong; MoE Key Laboratory of High Confidence Software Technologies, Hong Kong SAR, China

Rumor spread in social media severely jeopardizes the credibility of online content. Thus, automatic debunking of rumors is of great importance to keep social media a healthy environment. While facing a dubious claim, people often dispute its truthfulness sporadically in their posts containing various cues, which can form useful evidence with long-distance dependencies. In this work, we propose to learn discriminative features from microblog posts by following their non-sequential propagation structure and generate more powerful representations for identifying rumors. For modeling non-sequential structure, we first represent the diffusion of microblog posts with propagation trees, which provide valuable clues on how a claim in the original post is transmitted and developed over time. We then present a bottom-up and a top-down tree-structured models based on *Recursive Neural Networks* (RvNN) for rumor representation learning and classification, which naturally conform to the message propagation process in microblogs. To enhance the rumor representation learning, we reveal that effective rumor detection is highly related to finding evidential posts, e.g., the posts expressing specific attitude towards the veracity of a claim, as an extension of the previous RvNN-based detection models that treat every post equally. For this reason, we design discriminative attention mechanisms for the RvNN-based models to selectively attend on the subset of evidential posts during the bottom-up/top-down recursive composition. Experimental results on four datasets collected from real-world microblog platforms confirm that (1) our RvNN-based models achieve much better rumor detection and classification performance than state-of-the-art approaches; (2) the attention mechanisms for focusing on evidential posts can further improve the performance of our RvNN-based method; and (3) our approach possesses superior capacity on detecting rumors at a very early stage.

CCS Concepts: • **Computing methodologies** → **Artificial intelligence**; **Natural language processing**; • **Information systems** → **Web services**;

Additional Key Words and Phrases: Rumor detection and classification, social media, propagation tree, recursive neural networks, neural attention

Jing Ma work done when studying at the Chinese University of Hong Kong.

This work is partially supported by Hong Kong RGC GRF (14204118, 14209416) and ITF (ITS/335/18).

Authors' addresses: J. Ma, Department of Computer Science, 6F, David C. Lam Building, Hong Kong Baptist University, 55 Renfrew Road, Kowloon Tong, H.K; email: mmjjblue@gmail.com; W. Gao, School of Information Systems, Singapore Management University, 80 Stamford Road, Singapore 178902; email: weigao@smu.edu.sg; S. Joty, Nanyang Technological University, Block N4, 02c-79, Nanyang Avenue, Singapore, 639798; Salesforce Research, Singapore; email: srjoty@ntu.edu.sg; K.-F. Wong, Dept. of SEEM, The Chinese University of Hong Kong, Room 601, Ho Sin Hang Engineering Building, CUHK, Shatin, H.K.; MoE Key Laboratory of High Confidence Software Technologies, Hong Kong SAR, China; email: kfwong@se.cuhk.edu.hk.

2157-6904/2020/06-ART42 $15.00
https://doi.org/10.1145/3391250

ACM Reference format:
Jing Ma, Wei Gao, Shafiq Joty, and Kam-Fai Wong. 2020. An Attention-based Rumor Detection Model with Tree-structured Recursive Neural Networks. *ACM Trans. Intell. Syst. Technol.* 11, 4, Article 42 (June 2020), 28 pages.
https://doi.org/10.1145/3391250

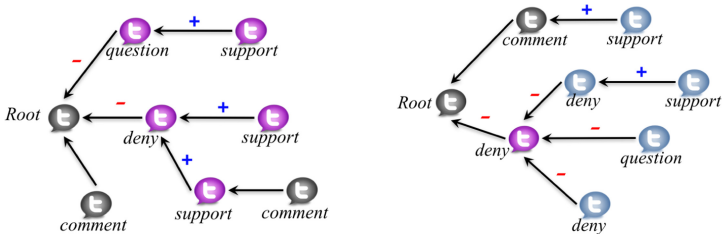
1 INTRODUCTION

Rumors have always been a social disease. Thanks to the popularity of social media outlets on Twitter, Facebook, and so on, it has become increasingly convenient for the "evil-doers" to create and disseminate rumors in massive scale with very low cost. The worst effect of false rumors could result in devastating consequences to the individual and/or society.

Research pertaining to rumors historically spans multiple disciplines, such as philosophy and humanities [8, 10], social psychology [1, 20, 40], political studies [2, 4], management science [9, 22], and recently computer science and artificial intelligence [5, 16, 33, 38, 39, 55]. Rumor is commonly defined as information that emerges and spreads among people whose truth value is unverified or intentionally false [8, 38]. Analysis shows that people tend to stop spreading a rumor if it is known as false [59]. However, identifying such misinformation is non-trivial and needs professional investigative journalism to fact-check the suspected claim, which is labor-intensive and time-consuming. The proliferation of social media content makes the situation worse due to the ever-increasing information load and dynamics. Therefore, it is necessary to develop automatic and assistant approaches to facilitate real-time rumor tracking and debunking.

Rumor debunking aims to determine the specific veracity of a given topic or a claim, and debunking rumors at an early stage is particularly crucial to minimizing their harmful effects. Journalists and online fact-checking websites such as snopes.com and politifact.com rely on manual effort to track and debunk rumors, which have obvious limitations on efficiency and topic coverage. For automating rumor detection, most of the previous studies focused on text mining from sequential microblog streams using supervised models based on feature engineering [5, 26, 27, 33]. However, such method requires heavy engineering effort that is painstakingly detailed and labor-intensive. For alleviating labor cost and better performance, Ma et al. [32] then proposed a data-driven sequential method using recurrent neural networks (RNN) to capture the dynamic temporal characteristics of rumor diffusion.

The method, however, oversimplifies the structural information associated with message propagation that can provide useful clues indicative of rumors. Propagation structures have been shown conducive to false rumors detection [50]. A model based on propagation tree kernel [34], which integrates information of post content, user, and propagation pattern, was then proposed to differentiate rumorous and non-rumorous claims via comparing the similarities among their propagation trees. But such kind of approach cannot directly classify a tree without pairwise comparison with all other trees, which imposes unnecessary overhead, and it cannot automatically learn any high-level compositional feature representations out of the noisy surface features. To the best of our knowledge, there is no neural representation learning approach based on propagation tree structures that has been attempted except our recent work [36], where we conducted a initial study by making use of tree-structured *recursive neural networks* (RvNN) for detecting and classifying rumors in social media. The RvNN-based model is a type of tree-structured neural network, in which the representation associated with each post is computed from its direct linked nodes following the tree edges. This article is a natural extension of our original work in Reference [36] by (1) selectively attending on most evidential posts during bottom-up/top-down recursion; and (2) conducting more thorough experiments and analyses on large-scale datasets in different languages.



(a) False rumor (b) True rumor

Fig. 1. Propagation trees of two rumorous source tweets. Nodes may express stances on their parent as commenting, supporting, questioning, or denying. The edge arrow indicates the direction from a response to its responded node, and the polarity is marked as "+" ("-") for support (denial). The same node color indicates the same stance on the veracity of root node (i.e., source tweet).

In this article, we first present our original neural rumor detection approach based on RvNN to bridge the content semantics and propagation clues. RvNN and its variants were originally used to compose phrase or sentence representation for syntactic and semantic parsing [43, 44]. Unlike parsing, where the input is a parse tree of an individual sentence, the input to our model is a propagation tree rooted from a source post, and each tree node is a responsive post instead of an individual word. The semantics of post content and the response relationship among the posts are jointly captured via the recursive feature learning process along the tree structure.

So, why can such a recursive neural model do better for this task? In general, analysis has found that Twitter could "self-correct" some inaccurate information as users share opinions, conjecture, and evidence. To illustrate our intuition, Figure 1 exemplifies the propagation trees of two rumors in our dataset, one being false and the other being true.[1] Structure-insensitive methods typically relying on the relative ratio of different stances in the text cannot do well when such aggregated relativity is unclear, as in this example. However, it can be seen that when a post denies a false rumor, it tends to spark supportive or affirmative replies confirming the denial; in contrast, denial to a true rumor tends to trigger questioning or denying utterances in the replies. This observation suggests a more general hypothesis that the repliers tend to disagree with (or question) who support a false rumor or deny a true rumor, and agree with who deny a false rumor or support a true rumor. Meanwhile, rather than directly responding to the source post (i.e., the root post), a reply is usually responsive to its immediate ancestor [30, 58], suggesting an obvious local characteristic of the interaction. The RvNN model naturally utilizes such structural properties for learning to capture the rumor indicative signals and strengthen the representations by recursively aggregating the signals along different branches.

In this article, we propose two variants based on the standard RvNN, i.e., a *bottom-up* (BU) model and a *top-down* (TD) model, which represent the propagation tree structure from two different angles to recursively visit the nodes and combine their representations following distinct directions. With such basic architectures, the node features can be hierarchically refined by the recursion following the tree structure. Consequently, it can be expected that the discriminative features will be embedded into the learned representations more effectively. However, a potential issue of this model is that all responding posts are treated equally during the recursion, which may amplify the noise in the tree-structured representation learning. For example, the commenting posts in Figure 1 should have been less important due to their weak opinions towards the source claim.

[1]False (true) rumor means the veracity of the rumorous claim turns out to be false (true) even though its truthfulness might be unclear initially.

Previous studies [35, 38] have found that rumor detection can benefit from taking into account the different stances expressed in responding posts. In this work, we introduce a novel method to improve our basic RvNN-based models by learning to automatically attend on those most evidential posts that express specific stances. Inspired by the success of neural attention, we propose specific attention mechanisms to encourage our model to be focused on such responsive posts in the tree during the bottom-up/top-down recursion.

We conduct extensive experiments on four real-world microblog datasets of different languages and demonstrate that (1) the proposed RvNN-based method yields outstanding improvements over the state-of-the-art baselines by large margin; (2) the attention on most evidential posts over the propagation tree is effective; and (3) our method performs particularly well on early rumor detection, which is crucial for timely intervention and debunking.

The contributions of our article are summarized as follows:

- To the best of our knowledge, this is the first study that deeply integrates both structure and content semantics based on tree-structured RvNN for detecting rumors from microblog posts.
- We propose two variants of RvNN models based upon the bottom-up and top-down tree structures to generate better integrated representations for a claim by capturing both structural and textual properties that signal the rumors.
- Based on the BU- and TD-RvNNs, we enhance our approach with specific attention mechanisms that select the subset of most evidential responsive posts during the recursion.
- We constructed four tree-structured microblog datasets, three of them consisting of English tweets responding to real-world claims gathered from rumor debunking websites and one composed of Chinese microblog posts collected from Sina Weibo community management center. On each dataset, our approach achieves superior performance over state-of-the-art baselines for both rumor classification and early detection tasks. We also make our resources publicly available.[2]

Our article is organized as follows: In Section 2, we briefly review the related work. In Section 3, we define the problem and the notations. Section 4 presents our proposed BU- and TD-RvNN models based on two propagation tree structures. Section 5 extends the two models with recursive attention mechanisms. Section 6 describes model training details. Sections 7 and 8 provide experimental settings and result analyses. Section 9 concludes our work with some possible directions for future research.

2 RELATED WORK

In this section, we provide a review of the research work on three different topics that are related to our study.

2.1 Rumor Detection

The literature related to rumor and fake news detection has been reviewed by several comprehensive surveys [25, 41, 42, 57]. We only briefly review some prior works closely related to ours.

Most previous automatic approaches for rumor detection [5, 27, 53] intended to learn a supervised classifier by utilizing a wide range of features crafted from post contents, user profiles, and propagation patterns. Subsequent studies were then conducted to engineer new features such as those representing rumor diffusion and cascades [13, 16] characterized by comments with links to debunking websites. Kwon et al. [26] introduced a time-series-fitting model based on the volume

[2] https://github.com/majingCUHK/Rumor_RvNN.

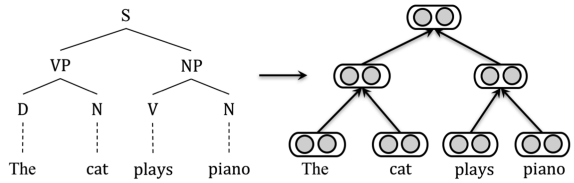


Fig. 2. A binarized sentence parse tree (left) and its corresponding RvNN architecture (right).

of tweets over time. Ma et al. [33] extended their model with a large set of chronological social context features. These approaches typically require heavy preprocessing and feature engineering.

Zhao et al. [55] alleviated the engineering effort by using a set of regular expressions (such as "really?," "not true," etc.) to find questing and denying tweets, but the approach was oversimplified and suffered from very low recall. Ma et al. [32] used recurrent neural networks (RNN) to learn automatically the representations from tweets content based on time series. Recently, they studied to mutually reinforce stance detection and rumor classification in a neural multi-task learning framework [35]. Guo et al. [15] proposed a hierarchical attention model that captures important clues from social context of a rumorous event at the post and sub-event levels. Jin et al. [21] exploited the conflicting viewpoints in a credibility propagation network for verifying news stories propagated among the tweets. However, these approaches cannot embed features reflecting how the posts are propagated and requires careful data segmentation to prepare for the time sequence.

Some kernel-based methods were exploited to model the propagation structure. Wu et al. [50] proposed a hybrid SVM classifier that combines an RBF kernel and a random-walk-based graph kernel to capture both flat and propagation patterns for detecting rumors on Sina Weibo. Ma et al. [34] used tree kernel to capture the similarity of propagation trees by counting their similar substructures to identify different types of rumors on Twitter. Compared to their studies, our model can learn the useful features via a more natural and general approach, i.e., the tree-structured neural network, to jointly generate the representation of a propagation tree based on the posts content and their propagation structure. Compared with the more recent tree-LSTM model [24] for rumor and stance detection that converted the propagation tree into a binarized constituency tree structure, our method directly takes as input a propagation tree and tries to learn tree representation, which is a more natural and accurate framework.

2.2 Recursive Neural Networks

RvNN is a type of tree-structured neural network. The original version of RvNN is built on the binarized sentence parse trees [43], in which the representation associated with each node of a parse tree is computed from its direct children. The overall structure of the standard RvNN is illustrated on the right side of Figure 2, corresponding to the input parse tree at the left side.

Generally, leaf nodes are the words in an input sentence, each represented by a low-dimensional word embedding. Non-leaf nodes are sentence constituents, computed by the recursion over the presentations of child nodes. Let p be the feature vector of a parent node whose children are c_1 and c_2, the representation of the parent is computed by:

$$p = f(W \cdot [c_1; c_2] + b), \tag{1}$$

where $f(\cdot)$ is the activation function with W and b as parameters. This computation is done recursively over all tree nodes; the learned hidden vectors of the nodes can then be used for various classification tasks.

In recent years, RvNN has demonstrated superior performance in a wide spectrum of tasks, e.g., images segmentation [44], phrase representation from word vectors [43], and sentiment

classification in sentences [45]. More recently, a deep RvNN was proposed to model the compositionality in natural language for fine-grained sentiment classification by stacking multiple recursive layers [19]. To avoid gradient vanishing, some studies integrated Long Short Term Memory (LSTM) [18] to RvNN [47, 56]. Mou et al. [37] used a convolutional network over tree structures for syntactic tree parsing of natural language sentences.

2.3 Attention Mechanism

Attention mechanism is in general described as: mapping a *query* and a set of *key-value* pairs into an *output*, where the *query*, *keys*, *values*, and *output* are all represented as low-dimensional vectors. More specifically, the output is computed as a weighted sum of the values, where the weight assigned to each value is computed by a compatibility function of the query with the corresponding keys. For example, in the context of Neural Machine Translation (NMT) task, the goal is to search for parts of the source sentence that are more relevant to predicting a target word. Let h_j denote the hidden vector of the jth word of the source sentence; to predict the ith word of the target sentence, the corresponding weight α_{ij} of each source word h_j is computed by:

$$\alpha_{ij} = \frac{\exp(\text{score}(s_{i-1}, h_j))}{\sum_{k=1}^{n} \exp(\text{score}(s_{i-1}, h_k))}, \tag{2}$$

where n denotes the number of source words, and the function score$(\cdot)$ evaluates how well the jth source word h_j (i.e., the "key" vector) matches with the $(i-1)$-th target prediction s_{i-1} (i.e., the "query" vector). An output vector c_i for improving NMT task is then computed as a weighted sum of all the source words (i.e., the "value" vector):

$$c_i = \sum_{j=1}^{n} \alpha_{ij} \cdot h_j. \tag{3}$$

In this way, the set of $\{\alpha_{ij}\}$ are weights defining how much each source word should be considered for predicting each target word; namely, only the selected regions from the source sentence are incorporated for further processing.

In recent years, attention-based neural network architectures, which learn to focus on their "attention" to specific parts of the input, have shown promising results on various tasks, such as parsing [11, 49], machine translation [31], image caption generation [52], natural language question answering [17, 23, 46, 51], and image question answering [29, 54].

Motivated from the successful applications in these tasks, we adopt attention mechanism to scan responsive posts and select those evidential ones that express specific stances for improving rumor detection. To the best of our knowledge, this is the first instance of applying attention in RvNN for rumor detection task.

3 PROBLEM STATEMENT AND NOTATIONS

On microblogging platforms such as Twitter, the follower/friend relationship embeds shared interests among the users. Once a user has posted a tweet, all his followers will receive it. Twitter allows a user to retweet or comment on another user's post, so that the information could reach beyond the followers of the original creator.

We model the propagation of each claim as a tree structure $\mathcal{T}(r) = \langle V, E \rangle$, where r is tree root representing the source tweet that states the claim, V refers to a set of nodes each representing a responding post of r (i.e., a retweet or a reply) in the thread of the circulation, and E is a set of directed edges corresponding to the response relation among the nodes in V.

An important issue of the tree structure is concerned about the direction of edges, which can result in two different architectures of the model: (1) a bottom-up tree and (2) a top-down tree, which are defined as follows:

- **Bottom-up tree** takes the similar shape as shown in Figure 1, where responding nodes always point to their responded nodes and the leaf nodes not having any response are laid out at the furthest level. For example, for any $u, v \in V$, $u \leftarrow v$ exists if v *responses to* u. This structure is similar to a citation network where a response mimics a reference.
- **Top-down tree** naturally conforms to the direction of information propagation, in which a link $u \rightarrow v$ means the information flows from u to v and v sees it and provides a response to u. This structure reverses bottom-up tree and simulates how information cascades from a source tweet, i.e., the root, to all its receivers, i.e., the decedents, which is similar as References [34, 50].

We define a rumor detection dataset as a set of claims $C = \{C_1, C_2, \ldots, C_{|C|}\}$, where each claim C_i corresponds to a source tweet r_i that consists of ideally all its relevant responding tweets in chronological order, i.e., $C_i = \{r_i, x_{i1}, x_{i2}, \ldots, x_{im}\}$ where each x_{i*} is a responding tweet of the r_i. Note that although the tweets are denoted sequentially, there are connections among them depending on their reply or repost relationships, which can form either a bottom-up tree or a top-down tree.

We formulate this task as a supervised classification problem that learns a classifier f from the labeled claims; that is, $f : C_i \rightarrow Y_i$, where Y_i takes one of the classes defined by the specific dataset, such as:

- Binary labels: *rumor* and *non-rumor*, which simply predicts a source claim as rumor or not;
- Finer-grained labels: *non-rumor*, *false rumor*, *true rumor*, and *unverified rumor*, which make rumor detection a more challenging classification problem [34, 59].

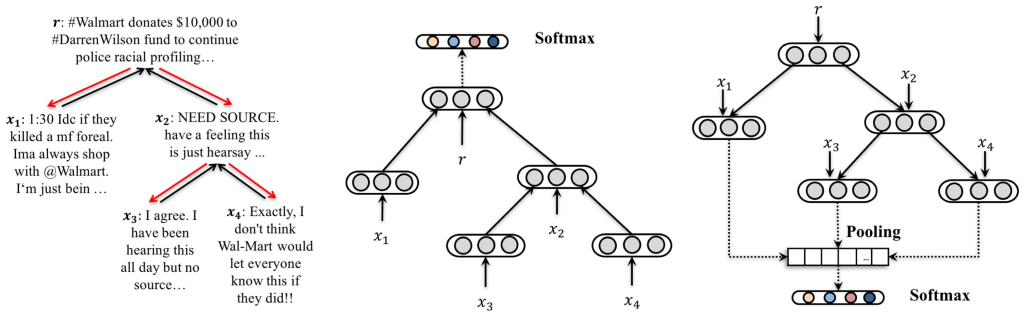
4 RVNN-BASED RUMOR DETECTION

The core idea of our method is to strengthen the high-level representation of tree nodes via a recursive learning process following the propagation structure over different branches in the tree. For instance, the responsive nodes confirming or supporting a node (e.g., "I agree," "be right," etc.) could further reinforce the stance of that node, while denial or questioning responses (e.g., "disagree," "really?!") otherwise would weaken its stance. Compared to the kernel-based method using propagation tree [34, 50], our method does not need pairwise comparison among large number of substructures and can learn a much stronger representation of content following the response structure.

In this section, we will describe our straightforward extension to the standard RvNN for modeling rumor detection based on the bottom-up and top-down architectures presented in Section 3. While the focus of our work is not detection by incorporating a wide variety of information, and we represent each tweet explicitly with only its textual content, this does not mean other type of information is completely ignored, as propagation structure can be implicitly embedded with the recursive learning process of the model.

4.1 Bottom-up RvNN

The core idea of bottom-up model is to generate a feature vector for each subtree by recursively visiting every node from the leaves at the bottom to the root at the top. In this way, the subtrees with similar contexts, such as those subtrees having a denial parent and a set of supportive children, will be projected into the proximity in the representation space. And thus such local rumor



(a) Bottom-up/Top-down tree (b) Bottom-up RvNN model (c) Top-down RvNN model

Fig. 3. A bottom-up/top-down propagation tree and the corresponding RvNN-based models. The black-color and red-color edges differentiate the bottom-up and top-down tree in Figure 3(a).

indicative features are aggregated along different branches into some global representation of the whole tree.

For this purpose, we make a natural extension to the original RvNN. The overall structure of our proposed bottom-up model is illustrated in Figure 3(b), taking a bottom-up tree (see Figure 3(a)) as input. Different from the standard RvNN, the input of each node in the bottom-up model is a post represented as a vector of words in the vocabulary in terms of tf-idf values. Here, every node has an input vector, and the number of children of nodes varies significantly.[3]

In rumor detection, long short-term memory (LSTM) [18] and gated recurrent units (GRU) [7] were used to learn textual representation, which adopts memory units to store information over long time steps [32]. In this article, we choose to extend GRU as hidden unit to model long-distance interactions over the tree nodes, because it is more efficient due to fewer parameters. Let $\mathcal{S}(j)$ denote the set of direct children of the node j. The transition equations of node j in the bottom-up model are formulated as follows:

$$\tilde{x}_j = x_j E, \tag{4}$$

$$h_{\mathcal{S}} = \sum_{s \in \mathcal{S}(j)} h_s, \tag{5}$$

$$\begin{aligned} r_j &= \sigma\left(W_r \tilde{x}_j + U_r h_{\mathcal{S}}\right), \\ z_j &= \sigma\left(W_z \tilde{x}_j + U_z h_{\mathcal{S}}\right), \\ \tilde{h}_j &= tanh\left(W_h \tilde{x}_j + U_h (h_{\mathcal{S}} \odot r_j)\right), \\ h_j &= (1 - z_j) \odot h_{\mathcal{S}} + z_j \odot \tilde{h}_j, \end{aligned} \tag{6}$$

where x_j is the original input vector of node j, E denotes the parameter matrix for transforming this input post, $\tilde{x}_j$ is the transformed representation of j, $[W_*, U_*]$ are the weight connections inside GRU, and h_j and h_s refer to the hidden state of j and its sth child. Thus, $h_{\mathcal{S}}$ denotes the sum of the hidden state of all the children of j assuming that all children are equally important to j. As with the standard GRU, $\odot$ denotes element-wise multiplication; a reset gate r_j determines how to combine the current input $\tilde{x}_j$ with the memory of children, and an update gate z_j defines how

[3]In standard RvNN, since an input instance is the parse tree of a sentence, only leaf nodes have input vector, each node representing a word of the input sentence, and the non-leaf nodes are constituents of the sentence, and thus the number of children of a node is limited.

much memory from the children is cascaded into the current node; and $\tilde{h}_j$ denotes the candidate activation of the hidden state of the current node. Different from the standard GRU unit, the gating vectors in our variant of GRU are dependent on the states of many child units, allowing our model to incorporate representations from different children.

After recursive aggregation from bottom to up, the state of root node (i.e., source tweet) can be regarded as the representation of the whole tree that is used for supervised classification. So, an output layer is connected to the root node for predicting the class of the tree using a softmax function:

$$\hat{y} = Softmax(Vh_0 + b), \tag{7}$$

where h_0 is the learned hidden vector of root node; V and b are the weights and bias in output layer.

4.2 Top-down RvNN

This model is designed to leverage the structure of top-down tree to capture complex propagation patterns for classifying claims, which is shown in Figure 3(c). It models how the information flows from source post to the current node. The idea of this top-down approach is to generate a strengthened feature vector for each post considering its propagation path, where rumor-indicative features are aggregated along the propagation history in the path. For example, if 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. Due to different branches led by any non-leaf node, the top-down visit to the subtree nodes of a non-leaf node is also recursive. However, the nature of top-down tree lends this model different from the bottom-up one. The representation of each node is computed by combining its own input and its parent node instead of its children nodes, which 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 yielding any loss. Then the hidden state h_j of a node j can be computed by combining the hidden state $h_{\mathcal{P}(j)}$ of its parent node $\mathcal{P}(j)$ and its own input vector x_j. Therefore, the transition equations of node j can be formulated as a standard GRU:

$$\tilde{x}_j = x_j E, \tag{8}$$

$$\begin{aligned} r_j &= \sigma\left(W_r \tilde{x}_j + U_r h_{\mathcal{P}(j)}\right), \\ z_j &= \sigma\left(W_z \tilde{x}_j + U_z h_{\mathcal{P}(j)}\right), \\ \tilde{h}_j &= tanh\left(W_h \tilde{x}_j + U_h (h_{\mathcal{P}(j)} \odot r_j)\right), \\ h_j &= (1 - z_j) \odot h_{\mathcal{P}(j)} + z_j \odot \tilde{h}_j. \end{aligned} \tag{9}$$

Through the top-down recursion, the learned representations are eventually embedded into the hidden vector of all the leaf nodes. Since the number of leaf nodes varies, the resulting vectors cannot be directly fed into a fixed-size neural layer for output. Therefore, we add a max-pooling layer to take the maximum value of each dimension of the vectors over all the leaf nodes. This can also help capture the most appealing indicative features from all the propagation paths.

Based on the pooling result, we finally use a softmax function in the output layer to predict the label of the tree:

$$\hat{y} = Softmax(Vh_\infty + b), \tag{10}$$

where h_∞ is the pooling vector over all leaf nodes, and V and b are parameters in the output layer.

Although both of the two RvNN models aim to capture the structural properties by recursively visiting all nodes, we can conjecture that the top-down model would be better. The hypothesis is that in the bottom-up case the final output relies on the representation of single root, and its information loss can be more evident than the top-down one, since in the top-down case the representations embedded into all leaf nodes along different propagation paths can be incorporated via pooling holistically.

5 ATTENTION-BASED RUMOR DETECTION MODEL

A serious defect of the above RvNN-based rumor detection models is that both bottom-up and top-down RvNNs treat all the tree nodes equally during the recursion. Intuitively, tweets are not all equal in a sense that the posts expressing fixed attitudes towards the source claim tend to be more evidential for rumor debunking. To illustrate this, we enclose additional responding posts to the source tweet in Figure 3(a) about "Walmart donates $10,000 to support Darren Wilson and the on going racist police murders," which are shown in Figure 4(a). We observe that the posts x_1–x_4 appear more evidential, as they express strong denial stance; x_5–x_6 are less useful, as they do not add anything to the veracity of the source claim. Thus, we utilize attention mechanisms to guide our model, paying more attention on the supposedly more important posts.

In this section, we present our attention-based method for improving the bottom-up and top-down RvNN. The structures of our proposed models are displayed in Figure 4(b) and 4(c), which takes either a bottom-up or a top-down tree (see Figure 4(a)) as input.

5.1 Attention-based Bottom-up RvNN

To strengthen the representation learning of the nodes, we guide the model to pay more emphasis on the evidential nodes using attention mechanism. To select the evidential posts, we manually create a "query" x_q as a surrogate of real-world post considering the text content of the post composed of a set of keywords/phrases that can indicate discriminative stance it holds (e.g., "true," "fake news," "really?"). As discussed in Section 2, a *query* should be a hidden vector. The representation of the "query" is thus generated using:

$$\tilde{x}_q = x_q E, \tag{11}$$

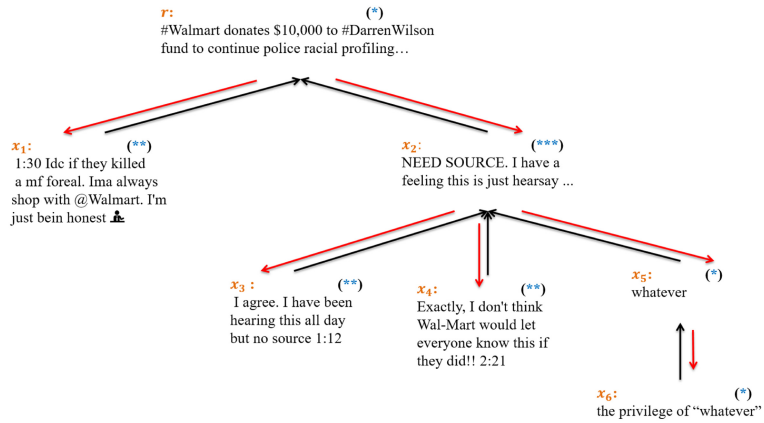
where x_q is the input vector of the "query" post and E is the matrix for transforming x_q. We then design two specific attention mechanisms based on the transformed representation of this "query" $\tilde{x}_q$.

Global attention is a straightforward extension based on the standard attention mechanism [23, 31], where we compute the attention weight indicating the relative likelihood of each node being evidential. Specifically, for each node $j \in V$, using the predefined GRU cell (see Equations (5)–(6)), we generate an l-dimensional hidden vector h_j. Then the attention weight of j is computed by[4]:

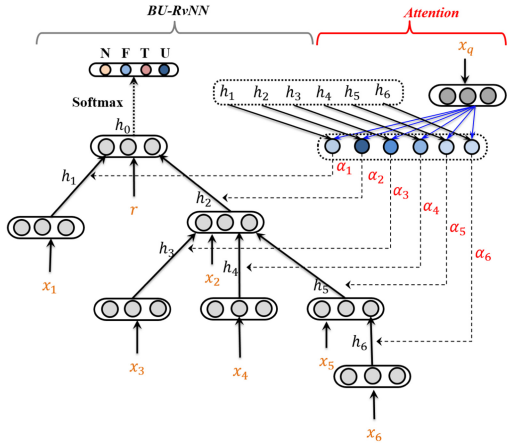
$$\begin{aligned} a_j &= \tanh((\tilde{x}_q \cdot W)^\top \cdot h_j), \\ \alpha_j &= \frac{\exp(a_j)}{\sum_{k=1}^{|V|} \exp(a_k)}, \end{aligned} \tag{12}$$

where $W \in \mathbb{R}^{l \times l}$ is the transformation parameter to compute an attention score a_j, α_j is the attention weight for node j, and $|V|$ denotes the number of nodes in the tree.

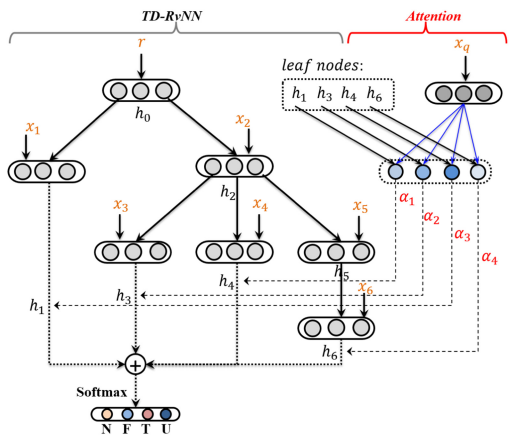
[4]Although we take a general form of dot product to compute the attention score [31], it is easy to be replaced with some recently proposed more sophisticated forms of operation such as additive attention [3], self-attention [6], scaled dot product attention [48], and so on.



(a) A bottom-up/top-down propagation tree



(b) Extension of bottom-up RvNN



(c) Extension of top-down RvNN

Fig. 4. Two attention-based structures of our proposed extension, taking a bottom-up/top-down tree (see Figure 4(a)) as the input. Here # of "*" indicates the relative importance of the post.

With the node vectors given by $\{h_j\}_{j=1}^{|V|}$ and their attention weights given by $\{\alpha_j\}_{j=1}^{|V|}$, we now combine them to represent the propagation tree. To generate an attention-based representation of the tree, we utilize a weighted summation of the hidden vectors of all tree nodes based on their corresponding attention weights:

$$h_G = \sum_{j=1}^{|V|} \alpha_j \cdot h_j. \tag{13}$$

Hierarchical Attention. Considering the subtree structure in each recursion step, we further develop a local attention model based on hierarchical attention mechanism that chooses to focus only on a small subset of the nodes in each recursion. Instead of computing the parent state by the summation over the hidden states of all its child nodes, we improve bottom-up RvNN by attending on those child states based on their likelihood of being evidential. This approach has an advantage of improving the representation of each node. Figure 4(b) illustrates its structure. The state value

for node j is calculated using Equation (15), and the attention weights for each of its child node $s \in \mathcal{S}(j)$ is given in Equation (14):

$$\begin{aligned} a_s &= \tanh((\tilde{x}_q \cdot W)^\top \cdot h_s), \\ \alpha_s &= \frac{\exp(a_s)}{\sum_{k \in \mathcal{S}(j)} \exp(a_k)}, \end{aligned} \tag{14}$$

$$\begin{aligned} \bar{h}_{\mathcal{S}} &= \sum_{s \in \mathcal{S}(j)} \alpha_s \cdot h_s, \\ \tilde{x}_j &= x_j E, \\ h_j &= \mathrm{GRU}(\tilde{x}_j, \bar{h}_{\mathcal{S}}, \theta), \end{aligned} \tag{15}$$

where the function GRU$(\cdot)$ is a shorthand of Equation (6). Similarity, $\tilde{x}_j$ is the input of node j, and E denotes the matrix weight for transforming. $\bar{h}_{\mathcal{S}}$ is the weighted child state, and θ represents all the parameters of GRU. The attention is done at the level of each subtree together with the bottom-up recursion in RvNN to compute the attention weights of upper-level subtrees, which thus forms a hierarchical attention process. Attention-based node representations are aggregated recursively up to the root node of the bottom-up tree whose state will be treated as the representation of the entire tree.

Differing from the original bottom-up RvNN described in Section 4.1, this structure decides how much information can flow from each child node to its parent node, taking into account the relative importance of each node being evidential. Therefore, the root node representation can be strengthened by the attention mechanism.

Finally, to predict the class of the tree, we process the final representation with a fully connected *Softmax* layer:

$$\begin{aligned} \text{Global Attention: } \hat{y} &= Softmax(Vh_G + b), \\ \text{Hierarchical Attention: } \hat{y} &= Softmax(V\bar{h}_0 + b), \end{aligned} \tag{16}$$

where $\bar{h}_0$ is the root node representation enhanced by the hierarchical attention mechanism.

5.2 Attention-based Top-down RvNN

Similarly, we improve the top-down RvNN by utilizing specific attention mechanisms to select the evidential posts.

Global attention. The global attention is built on top of the top-down recursion by attending over the whole set of nodes, which outputs the selected evidential nodes to the output layer. This shares the similar intuition as the global attention mechanism in the bottom-up model (see Section 5.1). More specifically, we first compute the representation for each node based on the predefined GRU cell (see Equation (9)) with our input posts. Then, we compute their attention weights by utilizing the "query" post. The top-down tree is finally represented as the weighted summation of the states of all the tree nodes.

Path attention. Unlike the bottom-up recursion, where each node is represented on the basis of aggregation of multiple child nodes, the hierachical attention cannot be readily applied to the top-down model, since each node is directly streamlined by its *only* parent node in the context of top-down recursion. Therefore, we design a specific attention mechanism, called "path attention," to attend on all the leaf nodes. Figure 4(c) illustrates the structure of this model.

Since the learned representation along each propagation path (i.e., from the root node to a leaf node) is eventually embedded into the hidden state of the corresponding leaf node, our intuitive idea is to selectively focus on the important paths in the sense that the path attention actually

takes into account not only the importance of the leaf nodes but also the information of all non-leaf nodes. For example, the path $r \rightarrow x_2 \rightarrow x_5 \rightarrow x_6$ in Figure 4(a) should be less important, because the involved nodes have higher proportion of commenting posts that do not contribute to assessing the veracity of the message, such as x_5 and x_6. Let $\mathcal{K}(r)$ denote the set of leaf nodes of the tree $\mathcal{T}(r)$; the attention weight for a path with node $k \in \mathcal{K}(r)$ as the endpoint is computed by:

$$\begin{aligned} b_k &= \tanh((\tilde{x}_q \cdot W) \cdot h_k^\top), \\ \beta_k &= \frac{\exp(b_k)}{\sum_{i \in \mathcal{K}(r)} \exp(b_i)}. \end{aligned} \tag{17}$$

Based on the attention weights $\{\beta_k\}$ obtained above, the top-down tree can be represented as the weighted sum of all leaf nodes, paying more attention on the key propagation paths with more evidential posts involved:

$$h_P = \sum_{k \in \mathcal{K}(V)} \beta_k \cdot h_k. \tag{18}$$

Finally, the tree representation obtained using the global attention or the path attention is fed into a *softmax* function to compute the probability distribution over the rumor classes.

Summary: Our attention mechanisms designed for the bottom-up/top-down RvNN differ in whether the "attention" is placed on the specific node sets after recursion or during each recursion step. That is, we can devise the hierarchical attention by attending over a subset of nodes during each recursive step due to the characteristic of bottom-up recursion, while the other three attention mechanisms only play a role at the output layer. Although all the attention mechanisms aim to select the important nodes, we can conjecture that: (1) in the bottom-up model, the hierarchical attention would be better, since each node aims to achieve a more accurate representation for each node during the recursion, but the global attention aims to only improve the final tree representation; and (2) in the top-down model, both of the two attentions are placed at the output layer, but the global attention could be better, because it attends on a wider range of nodes than only the leaf nodes in the path attention.

6 MODEL TRAINING

All our models are trained to minimize the squared error between the probability distribution of the prediction and that of the ground truth:

$$L(y, \hat{y}) = \sum_{n=1}^{N} \sum_{c=1}^{C} (y_c - \hat{y}_c)^2 + \lambda ||\Theta||_2^2 , \tag{19}$$

where y_c is the ground-truth label and $\hat{y}_c$ is the predicted probability of class c, N is the number of training claims, C is the number of classes, $||.||_2$ is the L_2 regularization term over all the model parameters Θ, and λ is the trade-off coefficient.

During training, all the model parameters are updated using efficient back-propagation through structure [14, 45]. The optimization algorithm is gradient-based following the Ada-grad update rule [12] to speed up the convergence. The learning rate initialized as 0.05. We empirically initialize the model parameters with uniform distribution and set the text-based vocabulary size as 5,000, the size of post text embedding and hidden units as 100. We iterate over all the training examples in each epoch and continue until the loss value converges or the maximum epoch number is met.

7 EXPERIMENTS AND RESULTS

In this section, we will describe the data collection as well as the comparative experiments and results.

7.1 Data Collection

We construct three microblog datasets based on data collection from Twitter[5] and Sina Weibo.[6] We collect a good number of source claims, which form the tree roots, together with their corresponding propagation trees. Each tree is appropriately annotated with a ground-truth label automatically obtained from the online rumor debunking services.

We constructed our Twitter datasets based on a couple of reference datasets, namely, Twitter15 [27] and Twitter16 [32]. The original datasets were released and used for a binary classification between rumors and non-rumors with respect to a set of given events that contain their relevant tweets. As a first step, we extracted popular source tweets that are frequently retweeted or replied. We then collected all the propagation threads (i.e., retweets and replies) for each of these source tweets. Because Twitter API cannot retrieve the retweets or replies, we gathered the retweet users for a given tweet from Twrench[7] and crawled the replies through Twitter's web interface. After the trees were constructed, we annotated the source tweets by referring to the labels of the events they were collected from. We first turned the label of each event in Twitter15 and Twitter16 from binary to the four-way veracity label (i.e., false rumor, true rumor, unverified rumor, and non-rumor) according to the original veracity tag of the claim corresponding to that event presented in the rumor debunking websites, including snopes.com , Emergent.info , and so on. We then labeled the source tweets by following these rules: (1) Source tweets from unverified rumor events or non-rumor events are labeled the same as the corresponding event's label; (2) For a source tweet in a false rumor event, we flip over the label and assign it as true to the source tweet if the tweet expresses denying or questioning stance; otherwise, the label is assigned as false; (3) The similar flip-over/no-change rule also applies to the source tweets in the true rumor events.

For the Weibo dataset, we obtain a set of verified rumors from the Sina's community management center,[8] which accepts user reports on various misinformation. The Weibo API can capture the original messages and all their repost/reply messages given a source post. We also gather a similar number of non-rumor claims by crawling the posts of general threads that are not reported as rumors. The resulting dataset consists of 2,313 rumors and 2,351 non-rumors.

Moreover, the fraction of different types of rumors are imbalanced in real-world. For example, the number of real news usually far exceeds that of false rumors. Therefore, we resort to another public benchmark rumor dataset PHEME[9] [60], which is unbalanced and collected based on five real-world breaking news items.

We make the three datasets collected by us publicly accessible.[10] Table 1 displays the basic statistics of the resulting datasets.

7.2 Settings and Protocols

For evaluation, we will make comprehensive comparisons between our proposed models and state-of-the-art baselines on rumor classification and early detection tasks.

- DT-Rank: Zhao et al. [55] proposed a Decision-Tree-based Ranking model to identify trending rumors by searching for inquiry phrases.

[5] www.twitter.com.
[6] www.weibo.com.
[7] https://twren.ch.
[8] http://service.account.weibo.com.
[9] https://figshare.com/articles/PHEME_dataset_of_rumours_and_non-rumours/4010619.
[10] Twitter15&16: https://www.dropbox.com/s/7ewzdrbelpmrnxu/rumdetect2017.zip?dl=0, Weibo: https://www.dropbox.com/s/46r50ctrfa0ur1o/rumdect.zip?dl=0.

Table 1. Statistics of the Datasets

Statistic	Twitter15	Twitter16	PHEME	Weibo
# of source tweets	1,490	818	5,802	4,664
# of tree nodes	76,351	40,867	30,376	3,805,656
# of non-rumors	374	205	3,830	2,351
# of false rumors	370	205	1,972	2,313
# of true rumors	372	205	–	–
# of unverified rumors	374	203	–	–
Avg. time length / tree	444 Hours	196 Hours	18 Hours	2,461 Hours
Avg. # of posts / tree	52	50	6	816
Max # of posts / tree	814	757	228	59,318
Min # of posts / tree	1	1	3	10

- **DTC** and **SVM-RBF**: The information credibility model using a Decision-Tree Classifier [5] and the SVM-based model with RBF kernel [53], both using hand-crafted features that are based on the overall statistics of the posts without considering temporal information.

- **RFC**: The Random Forest Classifier, which used three fitting parameters as temporal properties and a set of hand-crafted features based on user, linguistic, and structural properties [26].

- **SVM-TS**: A linear SVM classifier that uses time-series to model the variation of feature values over time based on a set of hand-crafted social context features [33].

- **SVM-BOW**: A naive baseline we built by representing the text content using bag-of-words model and using linear SVM for rumor classification.

- **CPCV**: A news verification model [21] based on a Credibility Propagation network by exploring the Conflicting Viewpoints in tweets.

- **SVM-TK** and **SVM-HK**: An SVM classifier that uses a Tree Kernel [34] and that uses a Hybrid Kernel [50], respectively, both of which try to capture propagation structure via kernel learning.

- **GRU-RNN** and **HAS-RNN**: A rumor detection model based on recurrent neural networks [32] with GRU units for learning rumor representations by modeling sequential structure of relevant posts, and a hierarchical attention network with a set of social features [15], which is an extension of non-attention-based GRU-RNN for rumor detection, respectively.

- **BU-RvNN** and **TD-RvNN**: Our proposed bottom-up and top-down RvNN models, respectively (see Section 4).

- **BU-RvNN-GA** and **TD-RvNN-GA**: Our extension model of BU-RvNN and TD-RvNN, respectively, both with the Global Attention mechanism (see Section 5).

- **BU-RvNN-HA** and **TD-RvNN-PA**: Our extension on the BU-RvNN model using Hierarchical Attention and the TD-RvNN model using Path Attention, respectively (see Section 5).

We implement **DT-Rank**, **DTC**, and **RFC** using Weka,[11] SVM-based models using LibSVM[12] and all neural-network-based models with Theano.[13] We use accuracy, AUC, and class-specific F-measure as evaluation metrics. We hold out 10% of the datasets for tuning the hyperparameters and conduct five-fold cross-validation on the rest of the datasets.

8 RESULTS AND ANALYSIS

We now provide a detailed description and analysis of the experimental results based on the three datasets in the following subsections.

[11] www.cs.waikato.ac.nz/ml/weka.
[12] www.csie.ntu.edu.tw/~cjlin/libsvm.
[13] deeplearning.net/software/theano.

Table 2. Results of Comparison among Different Models on Twitter15 and Twitter16 Datasets

Dataset	Twitter15						Twitter16					
Method			NR	FR	TR	UR			NR	FR	TR	UR
	Acc.	AUC	F_1	F_1	F_1	F_1	Acc.	AUC	F_1	F_1	F_1	F_1
DT-Rank	0.409	0.606	0.501	0.311	0.364	0.473	0.414	0.621	0.394	0.273	0.630	0.344
SVM-RBF	0.318	0.551	0.455	0.037	0.218	0.225	0.321	0.541	0.423	0.085	0.419	0.037
DTC	0.454	0.634	0.733	0.355	0.317	0.415	0.465	0.648	0.643	0.393	0.419	0.403
RFC	0.565	0.788	**0.810**	0.422	0.401	0.543	0.585	0.812	0.752	0.415	0.547	0.563
SVM-TS	0.544	0.780	0.796	0.472	0.404	0.483	0.574	0.804	**0.755**	0.420	0.571	0.526
SVM-BOW	0.548	0.754	0.564	0.524	0.582	0.512	0.585	0.777	0.553	0.556	0.655	0.578
CPCV	0.608	0.810	0.593	0.528	0.710	0.589	0.628	0.830	0.659	0.559	0.661	0.625
SVM-HK	0.493	0.717	0.650	0.439	0.342	0.336	0.511	0.736	0.648	0.434	0.473	0.451
SVM-TK	0.667	0.781	0.619	0.669	0.772	0.645	0.662	0.798	0.643	0.623	0.783	0.655
GRU-RNN	0.641	0.778	0.684	0.634	0.688	0.571	0.633	0.786	0.617	0.715	0.577	0.527
HAS-RNN	0.675	0.803	0.615	0.694	0.777	0.625	0.660	0.791	0.586	0.675	0.745	0.600
BU-RvNN	0.708	0.868	0.695	0.728	0.759	0.653	0.718	0.870	0.723	0.712	0.779	0.659
TD-RvNN	0.723	0.880	0.682	0.758	**0.821**	0.654	0.737	0.879	0.662	0.743	0.835	0.708
BU-RvNN-GA	0.726	0.889	0.673	0.738	0.809	**0.689**	0.741	0.884	0.720	0.725	0.797	0.699
TD-RvNN-GA	**0.756**	**0.913**	0.784	**0.774**	0.817	0.680	**0.764**	**0.896**	0.708	**0.753**	**0.840**	**0.738**
BU-RvNN-HA	0.739	0.902	0.752	0.733	0.797	0.676	0.755	0.890	0.719	0.748	0.800	0.727
TD-RvNN-PA	0.736	0.896	0.734	0.726	0.814	0.669	0.748	0.884	0.687	0.746	0.837	0.726

NR: non-rumor; FR: false rumor; TR: true rumor; UR: unverified rumor.

8.1 Twitter Datasets

As shown in Table 2, our proposed RvNN-based models basically yield much better performance than other methods on both datasets via the modeling of nonsequential structures of responsive posts in the propagation.

It is observed that the performances of the five baselines in the first group based on hand-crafted features are obviously poor, varying between 0.318 and 0.585 in accuracy on the two datasets, indicating that they fail to generalize due to the lack of capacity capturing helpful features. Among these baselines, SVM-TS and RFC perform relatively better, because they use additional temporal traits, but they are still clearly worse than the models not relying on feature engineering, which are in the other two groups. DTR uses a set of regular expressions indicative of rumor signals. However, only 19.6% and 22.2% of tweets in the two datasets contain strings covered by these regular expressions, rendering unsatisfactory results.

Among the two baselines using kernel methods in the second group, which require comparing the substructures among propagation trees, we observe that SVM-TK is much more effective than SVM-HK. There are two reasons: (1) SVM-HK was originally proposed and experimented on Sina Weibo [50], which may not be generalized well to Twitter data; (2) SVM-HK loosely couples two separate kernels: an RBF kernel based on hand-crafted features, plus a random walk-based kernel that relies on a set of pre-defined keywords for jumping over the nodes probabilistically. This may under-utilize the propagation information due to such oversimplified treatment of tree structure. In contrast, SVM-TK is an integrated kernel and can fully utilize the structure by comparing the trees based on both textual and structural similarities.

It appears that simply using bag-of-words is already a decent model evidenced as the fairly good performance of SVM-BOW, which is even higher than SVM-HK. This is because the features of SVM-HK are originally hand-crafted for binary classification (i.e., non-rumor vs. rumor), whereas it ignores the importance of indicative words or units that benefit finer-grained classification, which, however, can be captured more effectively by SVM-BOW. CPCV slightly improves

the accuracy compared with SVM-BOW, because CPCV exploits supporting and opposing relations among tweets. However, CPCV is a topic model-based method with a fully connected graph as input, which ignores the characteristics of propagation structure, thus performs worse than GRU-RNN and HSA-RNN.

The sequential neural model GRU-RNN performs slightly worse than SVM-TK but much worse than our recursive models in the third group, despite it being further improved by the HSA-RNN with a limited attention mechanism that uses some social features such as user profiles and propagation-related attributes of an event to attend on subevents. This is because RNN is basically a special case of the recursive NN, in which each non-leaf node has one child only. It has to rely on a linear chain as input, which missed valuable structural information. However, it does learn high-level features from the post content via hidden units of the neural model, while SVM-TK cannot, as it can only evaluate the trees' similarity based on the overlapping words among their subtrees. Our recursive models (in the third group) are not only inherently tree-structured but also take advantages of representation learning following the full propagation structure, which thus beat SVM-TK and HSA-RNN.

Among our first two recursive models, TD-RvNN outperforms BU-RvNN, which indicates that the bottom-up model may suffer from larger information loss than the top-down one. This verifies the hypothesis we made in Section 4.2 that the pooling layer in the top-down model can effectively select the more important features embedded into the leaf nodes.

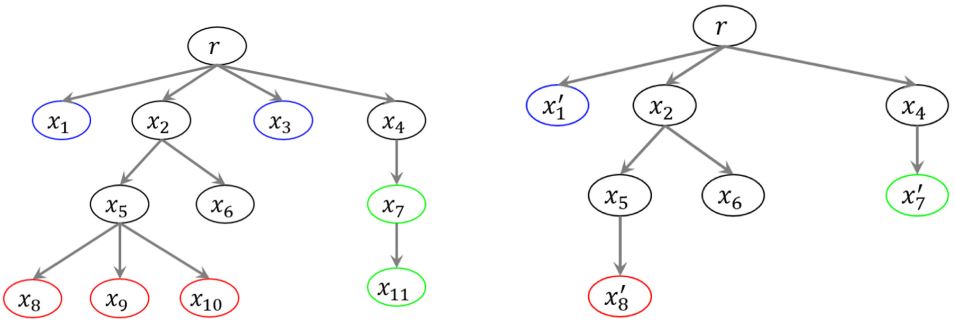
Our proposed extension of BU-/TD-RvNN using attention consistently outperforms their counterparts without attention, which suggests the effectiveness of our designed attention mechanisms for selecting the evidential posts. Specifically, in the bottom-up model, the hierachical attention makes higher improvement than the global attention does. The reason is that BU-RvNN-HA selectively focuses on the evidential posts at each recursive step and can optimize the representation of every node, while the attention method of BU-RvNN-GA only works at the output layer by keeping the representation of individual node intact. In the top-down model, we observe that the global attention gives higher improvement. This is because both of the mechanisms place the attention at the output layer but on a different range of nodes, i.e., the TD-RvNN-PA only attends on the leaf nodes, while the TD-RvNN-GA attends on all the nodes. Therefore, although BU-RvNN performs slightly worse than TD-RvNN, the attention-based model BU-RvNN-HA makes it perform comparably well as TD-RvNN-PA.

Furthermore, we observe that some feature-based baselines (e.g., RFC, SVM-TS) are clearly better on non-rumors. This can be explained from two aspects: (1) the features of these baseline models were originally defined for a binary (rumor vs. non-rumor) classification task. So, they do not perform well on finer-grained rumor classes; and (2) these models are trained with additional user features (e.g., profile, verification status) that may contain useful clues for differentiating non-rumors from rumors. We conjecture that the responses to non-rumors are usually much more diverse, which do not contain salient signals, making identification of non-rumors only based on content even more challenging.

8.1.1 Results on Unbalanced Dataset PHEME. The effectiveness of different models on the unbalanced dataset PHEME is comparatively shown in Table 3. We observe that all the feature engineering-based systems perform worse than the feature learning methods. In particular, our proposed models based on RvNN improve over the best baseline model HSA-RNN by 2.88%–7.27% in terms of accuracy, which is relatively lower than the improvements made on Twitter15 and Twitter16 datasets. This is because there are only six posts per claim on average, rendering limited structural information being available. However, the trend is consistent to what we have observed before, which confirms the effectiveness of our method even though the distribution of rumor and non-rumor classes is unbalanced and the number of posts per tree is much fewer.

Table 3. Results of Comparison among Different Models on PHEME Dataset

Method	Acc.	AUC	Rumor			Non-rumor		
			Prec.	Rec.	F_1	Prec.	Rec.	F_1
DT-Rank	0.657	0.588	0.472	0.239	0.317	0.695	0.867	0.772
SVM-RBF	0.656	0.595	0.318	0.539	0.401	0.688	0.842	0.758
DTC	0.670	0.657	0.572	0.435	0.494	0.687	0.837	0.755
RFC	0.709	0.712	0.637	0.285	0.393	0.778	0.843	0.809
SVM-TS	0.717	0.711	0.318	0.541	0.405	0.832	0.786	0.814
SVM-BOW	0.756	0.743	0.636	0.592	0.613	0.822	0.824	0.823
CPCV	0.763	0.809	0.699	0.447	0.548	0.815	0.862	0.839
SVM-HK	0.752	0.803	0.689	0.597	0.639	0.866	0.765	0.811
SVM-TK	0.785	0.752	0.657	0.689	0.677	0.851	0.827	0.839
GRU-RNN	0.775	0.823	0.667	0.643	0.658	0.825	0.840	0.832
HAS-RNN	0.797	0.854	0.703	0.630	0.663	0.854	0.857	0.855
BU-RvNN	0.820	0.861	0.722	**0.741**	0.731	**0.869**	0.857	0.867
TD-RvNN	0.829	0.866	0.761	0.717	0.736	0.861	0.887	0.873
BU-RvNN-GA	0.832	0.877	0.769	0.719	0.741	0.864	0.887	0.876
TD-RvNN-GA	**0.855**	**0.889**	**0.821**	0.719	**0.767**	0.868	0.922	**0.894**
BU-RvNN-HA	0.847	0.887	0.816	0.715	0.762	0.864	0.913	0.888
TD-RvNN-PA	0.841	0.880	0.769	0.697	0.731	0.842	**0.937**	0.887



(a) An original top-down Tree (b) A simplified top-down Tree

Fig. 5. A illustrative top-down tree before/after simplification. The nodes at left are merged into the ones at right in the same color. x'_i contains the post contents of all the merged nodes.

8.2 Weibo Dataset

On microblogs, a popular message can be reposted typically hundreds of thousands times, rendering extremely large propagation tree. In our experiments, when we tried the Weibo dataset, both BU-RvNN and TD-RvNN models training exceeded our GPU memory (12 GB) due to the large number of tree nodes in the dataset. We then realized that, because of recursive computation of the hidden states of nodes, the space complexity of our RvNN-based models is at least quadratic to the number of nodes, making them inefficient to handle large-scale trees.

To reduce the computation complexity, we develop an algorithm to simplify a tree structure by merging adjacent nodes, thus reducing the scale of trees as illustrated in Figure 5(a) and Figure 5(b).

Previous studies show that the widespread rumors usually attract the participation of influential users to promote the propagation [34]. Therefore, to retain these salient nodes, we propose to trim

ALGORITHM 1: Algorithm for simplifying a tree given the reference scale of RvNN-based model

Input: A tree needs to be simplified $\mathcal{T} = \langle V, E \rangle$,
Reference size N of RvNN-based model
Output: Simplified Tree $\mathcal{T} = \langle \hat{V}, \hat{E} \rangle$

/* Initialization */
$V_{leaf} = \{v_i | v_i \in V, v_i \rightarrow \varnothing\}$; $V_{parent} = \{v_j | v_j \in V, v_j \notin V_{leaf}\}$;
while *true* **do**
  /* Merging candidate leaf nodes into one node */
  Find $\bar{V}_{leaf} \subseteq V_{leaf}$ such that all nodes in $\bar{V}_{leaf}$ have a same parent v_p;
  Merge $\bar{V}_{leaf}$ into one node $\bar{v}$;
  Create new edge $(v_p \rightarrow \bar{v})$;
  $V \leftarrow remove(\bar{V}_{leaf})$;
  $V \leftarrow add(\bar{v})$;
  /* Merging candidate parent→leaf nodes into one node */
  Find $\tilde{v}_p \in V_{parent}$ such that $\tilde{v}_p$ has only one child node $\tilde{v}_c \in V_{leaf}$;
  Merge $\tilde{v}_p$ and $\tilde{v}_c$ into one node $\tilde{v}$;
  Replace node: $\tilde{v}_p = \tilde{v}$;
  $V \leftarrow remove(\tilde{v}_c)$;
  if $|V| > N$ **then**
    /* Update node sets: V_{leaf} and V_{parent} */
    $V_{leaf} \leftarrow remove(\bar{V}_{leaf} \cup \{\tilde{v}_c\})$;
    $V_{leaf} \leftarrow add(\{\bar{v}\} \cup \{\tilde{v}\})$;
    $V_{parent} \leftarrow remove(\tilde{v}_p)$;
  else
    /* Generate output */
    $\hat{V} = V, \hat{E} \leftarrow update(V, E)$;
    return $\mathcal{T} = \langle \hat{V}, \hat{E} \rangle$;
  end if
end while

a tree in an order from the bottom to the top. And we require that the scale of the resulting tree approximates a reference size N, which is pre-specified. Algorithm 1 describes the procedure for simplifying a tree. Initially, we divide all the nodes into two sets, i.e., leaf nodes and non-leaf nodes. Then, our system tries to discover candidate nodes that can be merged into a *super* node to reduce the tree size while minimizing the influence of such merge operation. We utilize the following rules for the simplification:

- Merge the leaf nodes that have a same parent such as x_8–x_{10} in Figure 5(a) into one node x'_8 in Figure 5(b). Such leaf nodes can be merged, because the shape of the tree (i.e., the height of the tree and the set of parent nodes) is retained, and their representations do not need to be refined during recursion, as they have no responsive posts, which suggests the content of leaf nodes is less influential. Such simplification minimally alters tree structure while keeping textual content intact. Therefore, despite the leaf nodes being merged, the influence is minimal, since conflicting stances from different leaf nodes of the same parent are cancelled out, which reduces to the simplified case where the stance of majority wins out.

Table 4. Results of Comparison among Different Models on Weibo Dataset

Method	Acc.	AUC	Rumor			Non-rumor		
			Prec.	Rec.	F_1	Prec.	Rec.	F_1
DT-Rank	0.732	0.728	0.738	0.715	0.726	0.726	0.749	0.737
SVM-RBF	0.818	0.813	0.822	0.812	0.817	0.815	0.824	0.819
DTC	0.831	0.829	0.847	0.815	0.831	0.815	0.847	0.830
RFC	0.849	0.843	0.786	0.959	0.864	0.947	0.739	0.830
SVM-TS	0.857	0.861	0.839	0.885	0.861	0.878	0.830	0.857
SVM-BOW	0.850	0.845	0.782	0.963	0.863	0.954	0.742	0.835
CPCV	0.876	0.871	0.822	0.935	0.880	0.943	0.811	0.872
SVM-HK	0.880	0.875	0.866	0.930	0.896	0.911	0.861	0.885
SVM-TK	0.902	0.892	0.855	**0.968**	0.908	**0.963**	0.836	0.895
GRU-RNN	0.899	0.911	0.865	0.946	0.904	0.940	0.852	0.894
HAS-RNN	0.905	0.966	0.875	0.957	0.910	0.939	0.862	0.899
BU-RvNN	0.928	0.953	0.914	0.951	0.932	0.934	0.905	0.919
TD-RvNN	0.932	0.956	0.921	0.961	0.941	0.932	0.912	0.921
BU-RvNN-GA	0.937	0.960	0.939	0.927	0.933	0.934	0.945	0.939
TD-RvNN-GA	**0.948**	**0.968**	0.944	0.952	**0.948**	0.952	0.944	**0.948**
BU-RvNN-HA	0.945	0.963	**0.963**	0.926	0.944	0.929	**0.964**	0.946
TD-RvNN-PA	0.939	0.960	0.946	0.930	0.938	0.932	0.947	0.940

- Merge a pair of nodes, such as x_7 and x_{11} in Figure 5(a), where one node is the parent of the other, which *is a leaf node and its only child*, into one node x'_7 as in Figure 5(b). Such pair can be merged, because the parent node attracts only one reply without further propagation through this path. Merging such pair of nodes could minimize the impact of the merge due to the simple context they form. And, since such linear chain of responses is very common in microblog data, the merge can significantly reduce the size of the tree.
- If the number of the resulting tree nodes is higher than N, the node reduction process continues; otherwise, it returns the resulting tree.

We emphasize that the worst case of our trimming algorithm is still consistent with exiting methods focusing on holistically capturing the "mainstream stances" from the wisdom of crowds, which is not very likely to happen (unless the tree structure is very simple or its size is very small) due to the strict condition in which the nodes could be merged, as stipulated above. Therefore, the merge operation would reduce a large number of nodes while not significantly changing the overall tree structure and the distribution of different stances in the tree.

8.2.1 Results. The events concerned in the Weibo dataset are labeled with binary classes, i.e., rumor and non-rumor, due to the specific annotation scheme adopted by Sina Weibo's Community Management Center. Table 4 shows the performance of all the compared systems. At first glance, it appears that the overall trend of performance is similar to that on the Twitter data. However, there are some noticeable differences.

Among all the baselines, SVM-TK, GRU-RNN, and HAS-RNN perform comparably well, which are obviously better than the other baseline models. This is because, for the binary classification, which is less challenging than the finer-grained classification, the representation learning algorithm and structural information have the similar capacity for detecting rumors. And SVM-HK, which is also focused on modeling propagation tree structure via kernel learning but designed specifically for Weibo data, performs relatively worse than SVM-TK but still clearly better than the

feature-engineering baseline approaches. This is because SVM-HK simply combines two kernels based on flat features (which are generally used in the feature-engineering models) with additional structural information, which cannot fully utilize the propagation structure.

In the last group of results, although we have simplified the structure of input trees that originally have large number of nodes, our proposed RvNN-based models and variants still outperform all the baselines. This is because our models can retain most information of the structure and content via the effective representation learning. Further improvements achieved by our attention-based proposals confirm the effectiveness of our method focusing on those evidential posts. Furthermore, TD-RvNN-GA and BU-RvNN-HA achieve the best and second best in accuracy, respectively, which is consistent with the results we have obtained on the three Twitter datasets.

8.3 Early Detection

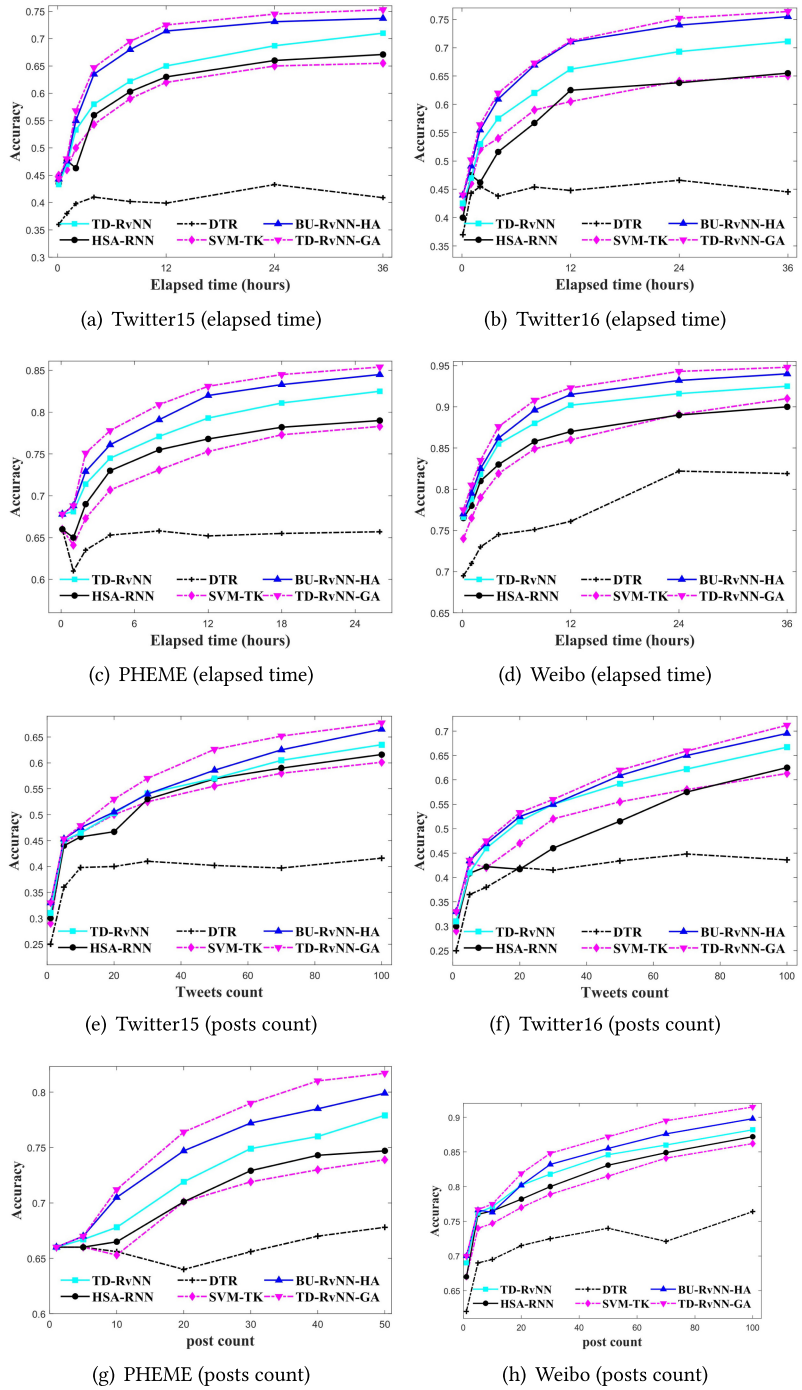
Debunking rumors at early stage of their propagation is very important so preventive measures can be taken in a timely manner. In early rumor detection task, we compare different detection methods at a series of checkpoints of "delays" that can be measured by either the count of responsive posts received or the time elapsed since the source claim was posted. The earlier the delay of a checkpoint is, the fewer propagation information is available at it. The performance is evaluated by the detection accuracy obtained at the specific checkpoint. To satisfy each checkpoint, we incrementally scan test data in order of time until the target time delay or post volume is reached.

Figure 6 shows the performance of our BU-RvNN-HA and TD-RvNN-GA models versus TD-RvNN (i.e., the best performed RvNN-based model without attention mechanism), SVM-TK (i.e., the best performed baseline with tree-structured input), HSA-RNN (i.e., the best performed baseline with sequential input), and DTR (i.e., an algorithm specifically proposed for early rumor detection) at various checkpoints.

We observe that within the first few hours, the performance of our recursive models grows more quickly and starts to supersede the other models at the early stage of propagation. Particularly, TD-RvNN-GA achieves 65.0% (75.5%) accuracy on Twitter15 (Twitter16) and 84.9% on PHEME within 24 hours, and TD-RvNN-GA achieves 94.5% on Weibo dataset within 12 hours, both of which demonstrate much faster detection rates than other models. Although all the methods are getting saturated as time goes by, BU-RvNN-HA and TD-RvNN-GA only need around 8 (6) hours or about 40 (20) tweets on Twitter15&16 and Weibo datasets (PHEME) to achieve the comparable performance of the best baseline model, i.e., HSA-RNN, which needs about 36 (24) hours or around 100 (50) posts on Twitter15&16 and Weibo datasets (PHEME), indicating superior early detection performance of our method. We also examine the impact of attention on early detection performance by comparing TD-RvNN with TD-RvNN-GA. We observe that TD-RvNN-GA clearly outperforms TD-RvNN especially when only limited propagation data is available, indicating that paying attention on evidential posts is particularly important to early detection and helpful.

8.4 Discussion

Thanks to one of our reviewers, we further studied a propagation-based early detection model named as PPC (Propagation Path Classification) [28], which outperforms our model on Twitter15&16 datasets. Different from our approach using post content, their method is based on user information. We believe that their higher performance lies in the following key factor: The users they used in the model include both reply users and retweet users, but the post contents we used are only from the reply users who have really posted their own words. Therefore, the number of user nodes in their model is far higher than that of the reply nodes in our model. Specifically, the number of user nodes in Twitter15 and Twitter16 datasets are 331,612 and 204,820, respectively (Table 1 in Reference [28]), while the number of content nodes used in our work are only 76,351



(a) Twitter15 (elapsed time)
(b) Twitter16 (elapsed time)
(c) PHEME (elapsed time)
(d) Weibo (elapsed time)
(e) Twitter15 (posts count)
(f) Twitter16 (posts count)
(g) PHEME (posts count)
(h) Weibo (posts count)

Fig. 6. Early rumor detection accuracy at different checkpoints in terms of elapsed time (or post count) on Twitter, PHEME, and Weibo datasets.

Table 5. Results of Comparison between Our RvNN-based Model with User Features and PPC

(a) Twitter Dataset

Dataset	Twitter15					Twitter16				
Method	Acc.	NR	FR	TR	UR	Acc.	NR	FR	TR	UR
		F_1	F_1	F_1	F_1		F_1	F_1	F_1	F_1
PPC	**0.842**	0.811	**0.875**	0.818	**0.790**	**0.863**	**0.820**	**0.898**	0.843	**0.837**
TD-RvNN-GA	0.756	0.784	0.774	0.817	0.680	0.764	0.708	0.753	0.840	0.738
TD-RvNN-GA(+U)	0.786	**0.822**	0.795	**0.831**	0.700	0.792	0.751	0.755	**0.864**	0.798

(b) Weibo Dataset

Method	Acc.	Rumor			Non-rumor		
		Prec.	Rec.	F_1	Prec.	Rec.	F_1
PPC	0.921	0.896	0.962	0.923	0.949	0.889	0.918
TD-RvNN-GA	0.948	0.944	0.952	0.948	0.952	0.944	0.948
TD-RvNN-GA(+U)	**0.961**	**0.961**	**0.962**	**0.961**	**0.962**	**0.961**	**0.962**

and 40,867 (Table 1). We only considered the posts containing user replies, since our method is motivated very differently from the PPC. Our conjecture can be confirmed by the comparable performances between their and our models on Weibo dataset due to the comparable numbers of user nodes and post nodes, since people cannot retweet on Weibo and all the posts have to contain user input.

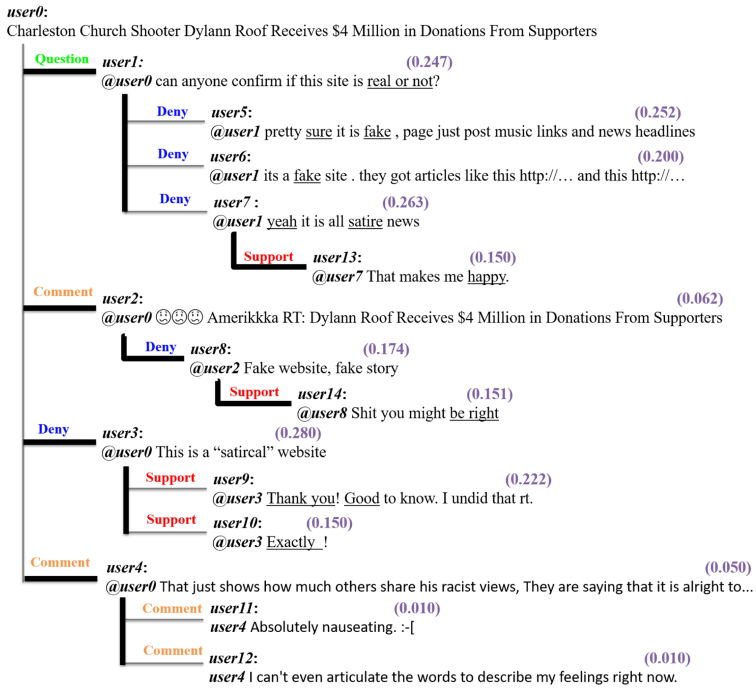
To reflect this with quantitative results, we enhance our RvNN-based models by judiciously incorporating user features. For each input post, we concatenate a set of normalized hand-crafted user features and the text embeddings of the post. Here, we choose TD-RvNN-GA, since it obtains superior performance than the other variants. We show the results in Table 5. We observe that TD-RvNN-GA(+U) outperforms PPC on Weibo dataset due to the similar number of tree nodes used in both models (because there is no retweet on Weibo). But PPC performs still better on Twitter15&16 datasets, confirming that the different number of tree nodes used really matters.

8.5 Case Study

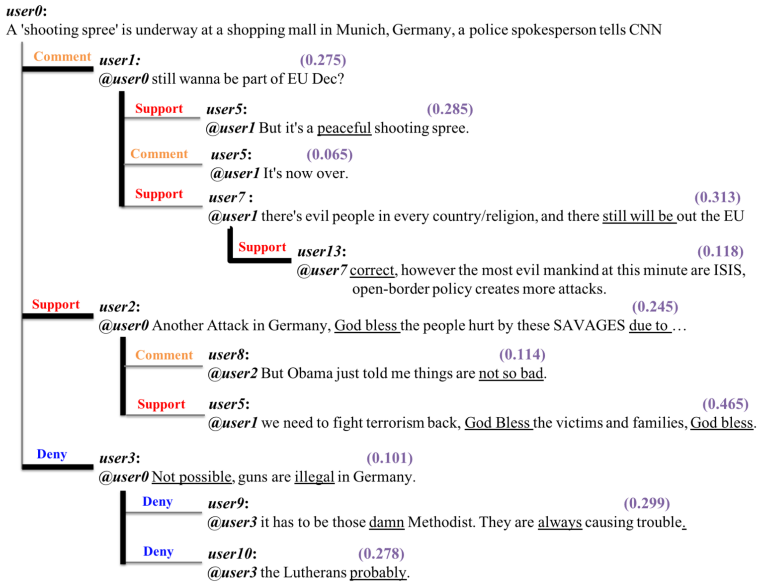
To get an intuitive understanding of what is happening when we use the RvNN-based model, we design an experiment to output the attention weights corresponding to each microblog post. Then, we display those posts together with their attention scores. Specifically, we choose to analyze the behavior of BU-RvNN-HA and TD-RvNN-GA, due to their superior performance compared with the other settings.

We sample a tree at the early stage of propagation for a claim about "Charleston Church Shooter Dylann Roof Receives $4 Million in Donations From Supporters," which has been correctly classified as a false rumor by both of the above models. In Figure 7(a), we can see that this false rumor demonstrates typical patterns in subtrees and propagation paths indicative of the falsehood, where a set of responses supporting the parent posts that deny (e.g., user3) or question (e.g., user1) the source post are captured by our bottom-up model. Similarly, some stance cascading patterns along the propagation path such as "deny→support" are also captured by our top-down model. In comparison, the non-structured and sequential models may be easily misled by the seemingly supportive key terms such as "be right," "yeah," "exactly!" that dominate the responses, and the tree kernel-based model SVM-TK may miss similar subtrees by just comparing the surface words.

We also sample a tree for the claim "A 'shooting spree' is underway at a shopping mall in Munich, Germany, a police spokeperson tells CNN," which is correctly classified as a true rumor by our models as shown in Figure 7(b). Clearly different stance cascading patterns are along the



(a) A false rumor



(b) A true rumor

Fig. 7. Examples of correctly detected rumors at early stage by our models, where propagation paths are marked with relevant stances. Note that edge direction is not shown, as it applies to either case. Here, the score in brackets indicates the average attention score obtained from our neural attention mechanisms.

propagation paths as compared to the false rumor, where "deny→deny," "support→support," and "comment→support→support" can be observed. This confirms the previous analysis that social media could "self-correct" some inaccurate information, e.g., when a post denies or questions a false rumor, it tends to spark supportive or affirmative replies confirming the objection, while denial to a true rumor tends to trigger denying utterances in the replies that deny the denial.

We also find that those highly ranked posts (indicated by the higher attention scores) play a major role in determining the class of the claim, since most of them are either expressing specific stances towards the claim. Our key motivation of introducing attention mechanisms into our RvNN-based models is to select such evidential posts. For example, the retweets that just repeat the root post and the posts that add no additional information seems less evidential. In contrast, the posts with specific stances, such as the denying posts created by *user3*, *user5,* and *user7* in Figure 7(a), and those supportive posts created by *user5* and *user7* and the denying posts by *user9* and *user10* in Figure 7(b), obtain higher attention scores, which are more likely embedded into the final representation for more accurate detection.

9 CONCLUSION AND FUTURE WORK

Existing rumor detection approaches often assume the dissemination of a source claim as flat sequential structure without making better use of the fact that the microblogs are networked data. In this article, we represent the spread of a hypothesis (i.e., a source claim) as a bottom-to-up tree and a top-to-down tree formed by harvesting users' responses triggered by the source claim. To encode the tree-structured data, we, respectively, propose a bottom-up and a top-down tree-structured model based on recursive neural networks for rumor detection on microblogging websites (e.g., Twitter and Sina Weibo). The inherent nature of recursive models allows them to use propagation tree to guide the learning of representations from post content, such as embedding various indicative features hidden in the structure, for better identifying rumors. Furthermore, we propose to strengthen the tree representation by developing novel attention mechanisms to concentrate on embedding evidential posts that may express certain stances towards the source claim. Results on three real-world datasets collected from Twitter and Sina Weibo show that: (1) our RvNN-based models yield outstanding rumor detection performance as compared to state-of-the art baselines; and (2) our extended models with specific attention over evidential tree nodes make large-margin improvements over the original RvNN-based models.

We will explore the following relevant issues in our future studies: (1) In this article, we only consider the textual information of the corresponding posts. In fact, other types of information such as user profiles, the friend/follower connections among users, post time, and so on, have shown usefulness for rumor detection. We will take advantage of more information types for improving rumor detection task. (2) Attention is a fundamental approach that has been studied heavily in recent years. Novel forms of attention mechanisms such as co-attention, self-attention, transformer, and so on, have obtained superior results in many NLP applications. An interesting and possible direction would be to use more effective attention mechanisms to further improve our recursive-based models.

REFERENCES

[1] G. W. Allport and L. J. Postman. 1965. *The Psychology of Rumor*. Russell & Russell.
[2] Gordon W. Allport and Leo Postman. 1946. An analysis of rumor. *Pub. Opin. Quart.* 10, 4 (1946), 501–517.
[3] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR'15)*.
[4] Adam J. Berinsky. 2017. Rumors and health care reform: Experiments in political misinformation. *Brit. J. Polit. Sci.* 47, 2 (2017), 241–262.

[5] Carlos Castillo, Marcelo Mendoza, and Barbara Poblete. 2011. Information credibility on Twitter. In *Proceedings of the World Wide Web Conference (WWW'11)*. 675–684.
[6] Jianpeng Cheng, Li Dong, and Mirella Lapata. 2016. Long short-term memory-networks for machine reading. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. 551–561.
[7] 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* (2014).
[8] Nicholas DiFonzo and Prashant Bordia. 2007. Rumor, gossip and urban legends. *Diogenes* 54, 1 (2007), 19–35.
[9] Nicholas DiFonzo, Prashant Bordia, and Ralph L. Rosnow. 1994. Reining in rumors. *Organ. Dyn.* 23, 1 (1994), 47–62.
[10] Pamela Donovan. 2007. How idle is idle talk? One hundred years of rumor research. *Diogenes* 54, 1 (2007), 59–82.
[11] Timothy Dozat and Christopher D. Manning. 2016. Deep Biaffine attention for neural dependency parsing. *CoRR* abs/1611.01734 (2016).
[12] John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *J. Mach. Learn. Res.* 12, July (2011), 2121–2159.
[13] Adrien Friggeri, Lada A. Adamic, Dean Eckles, and Justin Cheng. 2014. Rumor cascades. In *Proceedings of the International Conference on Weblogs and Social Media*.
[14] Christoph Goller and Andreas Kuchler. 1996. Learning task-dependent distributed representations by backpropagation through structure. In *Proceedings of the IEEE International Conference on Neural Networks*, Vol. 1. IEEE, 347–352.
[15] Han Guo, Juan Cao, Yazi Zhang, Junbo Guo, and Jintao Li. 2018. Rumor detection with hierarchical social attention network. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. ACM, 943–951.
[16] Aniko Hannak, Drew Margolin, Brian Keegan, and Ingmar Weber. 2014. Get back! You don't know me like that: The social mediation of fact checking interventions in Twitter conversations. In *Proceedings of the International Conference on Weblogs and Social Media*.
[17] Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Proceedings of the Conference on Advances in Neural Information Processing Systems*. 1693–1701.
[18] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neur. Comput.* 9, 8 (1997), 1735–1780.
[19] Ozan Irsoy and Claire Cardie. 2014. Deep recursive neural networks for compositionality in language. In *Proceedings of the 27th International Conference on Neural Information Processing Systems (NIPS'14)*, Vol. 2. 2096–2104.
[20] Marianne E. Jaeger, Susan Anthony, and Ralph L. Rosnow. 1980. Who hears what from whom and with what effect: A study of rumor. *Person. Soc. Psychol. Bull.* 6, 3 (1980), 473–478.
[21] Zhiwei Jin, Juan Cao, Yongdong Zhang, and Jiebo Luo. 2016. News verification by exploiting conflicting social viewpoints in microblogs. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence*. AAAI Press, 2972–2978.
[22] Allan J. Kimmel. 2004. *Rumors and Rumor Control: A Manager's Guide to Understanding and Combatting Rumors*. Routledge.
[23] Ankit Kumar, Ozan Irsoy, Peter Ondruska, Mohit Iyyer, James Bradbury, Ishaan Gulrajani, Victor Zhong, Romain Paulus, and Richard Socher. 2016. Ask me anything: Dynamic memory networks for natural language processing. In *Proceedings of the International Conference on Machine Learning*. 1378–1387.
[24] Sumeet Kumar and Kathleen M. Carley. 2019. Tree LSTMs with convolution units to predict stance and rumor veracity in social media conversations. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. 5047–5058.
[25] Srijan Kumar and Neil Shah. 2018. False information on web and social media: A survey. *arXiv preprint arXiv:1804.08559* (2018).
[26] Sejeong Kwon, Meeyoung Cha, Kyomin Jung, Wei Chen, and Yajun Wang. 2013. Prominent features of rumor propagation in online social media. In *Proceedings of the IEEE International Conference on Data Mining*. 1103–1108.
[27] 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 Conference on Information and Knowledge Management (CIKM'15)*. 1867–1870.
[28] 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 *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*. 354–361.
[29] Jiasen Lu, Jianwei Yang, Dhruv Batra, and Devi Parikh. 2016. Hierarchical question-image co-attention for visual question answering. In *Proceedings of the Conference on Advances in Neural Information Processing Systems*. 289–297.
[30] Michal Lukasik, P. K. Srijith, Duy Vu, Kalina Bontcheva, Arkaitz Zubiaga, and Trevor Cohn. 2016. Hawkes processes for continuous time sequence classification: An application to rumour stance classification in Twitter. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. 393–398.

[31] Minh-Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP'15)*. 1412–1421.

[32] 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 *Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI'16)*. 3818–3824.

[33] 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 Conference on Information and Knowledge Management (CIKM'15)*. 1751–1754.

[34] 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)*. 708–717.

[35] Jing Ma, Wei Gao, and Kam-Fai Wong. 2018. Detect rumor and stance jointly by neural multi-task learning. In *Proceedings of the the World Wide Web Conference (WWW'18)*. 585–593.

[36] 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)*. 1980–1989.

[37] Lili Mou, Hao Peng, Ge Li, Yan Xu, Lu Zhang, and Zhi Jin. 2015. Discriminative neural sentence modeling by tree-based convolution. *arXiv preprint arXiv:1504.01106* (2015).

[38] Vahed Qazvinian, Emily Rosengren, Dragomir R. Radev, and Qiaozhu Mei. 2011. Rumor has it: Identifying misinformation in microblogs. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP'11)*. 1589–1599.

[39] Jacob Ratkiewicz, Michael Conover, Mark Meiss, Bruno Gonçalves, Snehal Patil, Alessandro Flammini, and Filippo Menczer. 2011. Truthy: Mapping the spread of astroturf in microblog streams. In *Proceedings of the 20th International World Wide Web Conference (WWW'11)*. 249–252.

[40] Ralph L. Rosnow and Eric K. Foster. 2005. Rumor and gossip research. *Psychol. Sci. Agenda* 19, 4 (2005).

[41] Karishma Sharma, Feng Qian, He Jiang, Natali Ruchansky, Ming Zhang, and Yan Liu. 2019. Combating fake news: A survey on identification and mitigation techniques. *arXiv preprint arXiv:1901.06437* (2019).

[42] Kai Shu, Amy Sliva, Suhang Wang, Jiliang Tang, and Huan Liu. 2017. Fake news detection on social media: A data mining perspective. *ACM SIGKDD Explor. Newslett.* 19, 1 (2017), 22–36.

[43] Richard Socher, Brody Huval, Christopher D. Manning, and Andrew Y. Ng. 2012. Semantic compositionality through recursive matrix-vector spaces. In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL'12)*. 1201–1211.

[44] Richard Socher, Cliff C. Lin, Chris Manning, and Andrew Y. Ng. 2011. Parsing natural scenes and natural language with recursive neural networks. In *Proceedings of the 28th International Conference on Machine Learning (ICML'11)*. 129–136.

[45] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. 1631–1642.

[46] Sainbayar Sukhbaatar, Jason Weston, Rob Fergus et al. 2015. End-to-end memory networks. In *Proceedings of the Conference on Advances in Neural Information Processing Systems*. 2440–2448.

[47] Kai Sheng Tai, Richard Socher, and Christopher D. Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. *arXiv preprint arXiv:1503.00075* (2015).

[48] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the Conference on Advances in Neural Information Processing Systems*. 5998–6008.

[49] Oriol Vinyals, Łukasz Kaiser, Terry Koo, Slav Petrov, Ilya Sutskever, and Geoffrey Hinton. 2015. Grammar as a foreign language. In *Proceedings of the Conference on Advances in Neural Information Processing Systems*. 2773–2781.

[50] Ke Wu, Song Yang, and Kenny Q. Zhu. 2015. False rumors detection on Sina Weibo by propagation structures. In *Proceedings of the IEEE 31st International Conference on Data Engineering (ICDE'15)*. IEEE, 651–662.

[51] Caiming Xiong, Victor Zhong, and Richard Socher. 2016. Dynamic coattention networks for question answering. *CoRR* abs/1611.01604 (2016).

[52] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *Proceedings of the International Conference on Machine Learning*. 2048–2057.

[53] Fan Yang, Yang Liu, Xiaohui Yu, and Min Yang. 2012. Automatic detection of rumor on Sina Weibo. In *Proceedings of the ACM SIGKDD Workshop on Mining Data Semantics (MDS'12)*. 13:1–13:7.

[54] Zichao Yang, Xiaodong He, Jianfeng Gao, Li Deng, and Alex Smola. 2016. Stacked attention networks for image question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 21–29.

[55] 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 (WWW'15)*. 1395–1405.

[56] Xiaodan Zhu, Parinaz Sobihani, and Hongyu Guo. 2015. Long short-term memory over recursive structures. In *Proceedings of the 32nd International Conference on Machine Learning*. 1604–1612.

[57] 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 (2018), 32.

[58] Arkaitz Zubiaga, Elena Kochkina, Maria Liakata, Rob Procter, and Michal Lukasik. 2016. Stance classification in rumours as a sequential task exploiting the tree structure of social media conversations. In *Proceedings of the 26th International Conference on Computational Linguistics: Technical Papers (COLING'16)*. 2438–2448.

[59] Arkaitz Zubiaga, Maria Liakata, Rob Procter, Geraldine Wong Sak Hoi, and Peter Tolmie. 2016. Analysing how people orient to and spread rumours in social media by looking at conversational threads. *PloS One* 11, 3 (2016), e0150989.

[60] Arkaitz Zubiaga, Geraldine Wong Sak Hoi, Maria Liakata, and Rob Procter. 2016. PHEME dataset of rumours and non-rumours. In *Figshare. Dataset*.

Received August 2019; revised March 2020; accepted March 2020