Detecting Rumors from Microblogs with Recurrent Neural Networks

Jing Ma[1] **Wei Gao**[2] **Prasenjit Mitra**[2] **Sejeong Kwon**[3] **Bernard J. Jansen**[2]
Kam-Fai Wong[1] **Meeyoung Cha**[3]
[1]The Chinese University of Hong Kong, Hong Kong SAR
[2]Qatar Computing Research Institute, Hamad Bin Khalifa University, Qatar
[3]Graduate School of Culture Technology, Korea Advanced Institute of Science and Technology, Korea
[1]{majing,kfwong}@se.cuhk.edu.hk, [2]{wgao,pmitra,bjansen}@qf.org.qa, [3]{gsj1029,meeyoungcha}@kaist.ac.kr

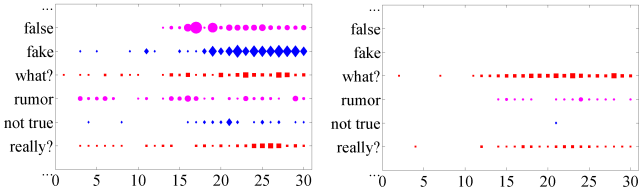
Abstract

Microblogging platforms are an ideal place for spreading rumors and automatically debunking rumors is a crucial problem. To detect rumors, existing approaches have relied on hand-crafted features for employing machine learning algorithms that require daunting manual effort. Upon facing a dubious claim, people dispute its truthfulness by posting various cues over time, which generates long-distance dependencies of evidence. This paper presents a novel method that learns continuous representations of microblog events for identifying rumors. The proposed model is based on recurrent neural networks (RNN) for learning the hidden representations that capture the variation of contextual information of relevant posts over time. Experimental results on datasets from two real-world microblog platforms demonstrate that (1) the RNN method outperforms state-of-the-art rumor detection models that use hand-crafted features; (2) performance of the RNN-based algorithm is further improved via sophisticated recurrent units and extra hidden layers; (3) RNN-based method detects rumors more quickly and accurately than existing techniques, including the leading online rumor debunking services.

1 Introduction

Social psychology literature defines a rumor as a story or a statement whose truth value is unverified or deliberately false [Allport and Postman, 1965]. False rumors are damaging as they cause public panic and social unrest. For example, on August 25th of 2015, a rumor about "shootouts and kidnappings by drug gangs happening near schools in Veracruz" spread through Twitter and Facebook[1]. This caused severe chaos in the city involving 26 car crashes, because people left their cars in the middle of a street and rushed to pick up their children from school. This incident of a false rumor highlights that automatically predicting the veracity of information on social media is of high practical value.

[1]http://www.bbc.com/news/world-latin-america-14800200



(a) In rumors (b) In non-rumors

Figure 1: Some shallow patterns signaling rumors along events' timelines (in # of hours since initial tweets), where the size of the shapes is the strength of their relative frequency

Debunking rumors at an early stage of diffusion is particularly crucial to minimizing their harmful effects. To distinguish rumors from factual events, individuals and organizations often have relied on common sense and investigative journalism. Rumor reporting websites like `snopes.com` and `factcheck.org` are such collaborative efforts. However, because manual verification steps are involved in such efforts, these websites are not comprehensive in their topical coverage and also can have long debunking delay.

Existing rumor detection models use learning algorithms that incorporate a wide variety of features manually crafted from the content, user characteristics, and diffusion patterns of the posts [Castillo *et al.*, 2011; Yang *et al.*, 2012; Kwon *et al.*, 2013; Liu *et al.*, 2015; Ma *et al.*, 2015; Wu *et al.*, 2015], or simply exploited patterns expressed using regular expressions to discover rumors in tweets [Zhao *et al.*, 2015]. Feature engineering is critical, but it is painstakingly detailed, biased, and labor-intensive. For example, two time series plots in Figure 1 depict the typical shallow patterns of rumor signals from [Zhao *et al.*, 2015]. Although they could demonstrate temporal traits of rumor and non-rumor events, differences between the two cases are neither clear-cut nor strong for feature engineering.

On the other hand, deep neural networks have demonstrated clear advantages for many machine learning problems [Sutskever *et al.*, 2011; 2014; Devlin *et al.*, 2014; Kombrink *et al.*, 2011; Cho *et al.*, 2014]. In this research, we explore opportunities to automatically discover and exploit deep data representations for efficient rumor detection. We posit that given the sequential nature of text streams in social media, recurrent neural networks (RNN) are suitable

for rumor detection. This is because the connections between units in an RNN form a direct cycle and create an internal state of the network [LeCun *et al.*, 2015, Figure 5] that might allow it to capture the dynamic temporal signals characteristic of rumor diffusion.

Utilizing RNN, we model the social context information of an event as a variable-length time series. We assume people, when exposed to a rumor claim, will forward the claim or comment on it, thus creating a continuous stream of posts. This approach learns both the temporal and textual representations from rumor posts under supervision. Extensive experiments on two real-world microblog datasets demonstrate that the RNN-based method yields outstanding performance. The model is also shown to be effective for *early detection* of rumors, where adequate accuracy could be achieved just several hours after the initial propagation.

The contributions of this paper are threefold:

- To the best of our knowledge, this is a first study using a deep learning model for rumor detection on microblogs. RNN-based model achieves significant improvements over state-of-the-art learning algorithms that rely on on hand-crafted features. The model is furhter extensible and can detect rumors more accurately than existing methods via sophisticated recurrent units and extra hidden layers.
- The RNN-based model further allows for early detection, demonstrating a clear efficacy when compared to existing baselines such as leading online rumor debunking services.
- We constructed two microblog datasets with ground truth labels for the task, in total containing more than 5,000 claims that scale to five million relevant microblog posts. We make this large rumor dataset fully public to be used for research purposes[2].

2 Related Work

Automatic rumor detection from social media is based on traditional classifiers that detect misinformation stemming from the pioneering study of information credibility on Twitter [Castillo *et al.*, 2011]. In following works [Yang *et al.*, 2012; Liu *et al.*, 2015; Ma *et al.*, 2015; Wu *et al.*, 2015], different sets of hand-crafted features were proposed and incorporated to determine whether a claim about an event is credible. Most of these prior works attempted to classify the veracity of spreading memes using information other than the text content, for instance, the popularity of a post (e.g., the number of retweets or replies of the post), the features relevant to determine a user's credibility, etc. However, feature engineering is painstakingly labor intensive. Our RNN-based method disregards this completely, yet can achieve better performance due to the effective representation learning capacity of deep neural models.

Some prior studies focused on capturing the temporal traits of rumors during their propagation. Kwon et al. [2013] introduced a time-series-fitting model based on the temporal properties of a single feature – tweet volume. Ma, et al. [2015] extended the model using dynamic time series to capture the variation of a set of social context features over time. Friggeri, et al. [2014] characterized the structure of misinformation cascades on Facebook by analyzing comments with links to rumor debunking websites. We also use the temporal properties of the representations, but the features are learned automatically via an RNN given the fundamental term representation in each time segment.

Zhao, et al. [2015] worked on early rumor detection using cue terms such as "not true", "unconfirmed" or "debunk" to find questioning and denying tweets. Our RNN learns representations that are significantly more complex than these explicit signals; these representations can capture the hidden implications and dependencies over time.

Our work is also related to, but different from, studies detecting spammers [Lee *et al.*, 2013; Wang, 2010; Hu *et al.*, 2013] and fake images on Twitter [Gupta *et al.*, 2013], and the "Truthy" system [Ratkiewicz *et al.*, 2011a; 2011b] that differentiates whether a meme is spreading organically or is being spread by an "astroturf" campaign.

3 RNN: Recurrent Neural Network

An RNN is a type of feed-forward neural network that can be used to model variable-length sequential information such as sentences or time series. A basic RNN is formalized as follows: given an input sequence $(x_1, \ldots, x_T)$, for each time step, the model updates the hidden states $(h_1, \ldots, h_T)$ and generates the output vector $(o_1, \ldots, o_T)$, where T depends on the length of the input. From $t = 1$ to T, the algorithm iterates over the following equations:

$$\begin{aligned} h_t &= tanh(Ux_t + Wh_{t-1} + b) \\ o_t &= Vh_t + c \end{aligned} \tag{1}$$

where *U*, *W* and *V* are the input-to-hidden, hidden-to-hidden and hidden-to-output weight matrices, respectively (see Figure 2(a) for the parameters of basic tanh-RNN), *b* and *c* are the bias vectors, and $tanh(.)$ is a hyperbolic tangent nonlinearity function.

Typically, the gradients of RNNs are computed via back-propagation through time [Rumelhart *et al.*, 1986]. In practice, because of the vanishing or exploding gradients [Bengio *et al.*, 1994], the basic RNN cannot learn long-distance temporal dependencies with gradient-based optimization. One way to deal with this is to make an extension that includes "memory" units to store information over long time periods, commonly known as Long Short-Term Memory (LSTM) unit [Hochreiter and Schmidhuber, 1997; Graves, 2013] and Gated Recurrent unit (GRU) [Cho *et al.*, 2014]. Here, we briefly introduce the two structures.

3.1 Long Short-Term Memory (LSTM)

Unlike the traditional recurrent unit whose state is overwritten at each time step (equations 1), an LSTM unit maintains a memory cell c_t at time t. The output h_t of an LSTM unit is computed by the following equations [Hochreiter and

[2] http://alt.qcri.org/~wgao/data/rumdect.zip

Schmidhuber, 1997; Graves, 2013]:

$$
\begin{aligned}
i_t &= \sigma(x_t W_i + h_{t-1} U_i + c_{t-1} V_i) \\
f_t &= \sigma(x_t W_f + h_{t-1} U_f + c_{t-1} V_f) \\
\tilde{c_t} &= tanh(x_t W_c + h_{t-1} U_c) \\
c_t &= f_t c_{t-1} + i_t \tilde{c_t} \\
o_t &= \sigma(x_t W_o + h_{t-1} U_o + c_t V_o) \\
h_t &= o_t \, tanh(c_t)
\end{aligned}
$$

where σ is a logistic sigmoid function. The input gate i_t determines the degree to which the new memory is added to the memory cell. The forget gate f_t decides the extent to which the existing memory is forgotten. The memory c_t is updated by forgetting part of the existing memory and adding new memory $\tilde{c_t}$. The output gate o_t is the amount of output memory.

3.2 Gated Recurrent Unit (GRU)

Similar to an LSTM unit, a GRU has gating units that modulate the flow of the content inside the unit, but a GRU is simpler with fewer parameters. The following equations are used for a GRU layer [Cho *et al.*, 2014]:

$$
\begin{aligned}
z_t &= \sigma(x_t U_z + h_{t-1} W_z) \\
r_t &= \sigma(x_t U_r + h_{t-1} W_r) \\
\tilde{h}_t &= tanh(x_t U_h + (h_{t-1} \cdot r_t) W_h) \\
h_t &= (1 - z_t) \cdot h_{t-1} + z_t \cdot \tilde{h}_t
\end{aligned}
$$

where a reset gate r_t determines how to combine the new input with the previous memory, and an update gate z_t defines how much of the previous memory is cascaded into the current time step, and $\tilde{h}_t$ denotes the candidate activation of the hidden state h_t.

4 RNN-based Rumor Detection

We present the details of our RNN-based model for classifying microblog events into rumors and non-rumors. First, we introduce a method that converts the incoming streams of microblog posts as continuous variable-length time series, and then describe RNNs with different kinds of hidden units and layers for classification.

4.1 Problem Statement

Individual microblog posts are short in nature, containing very limited context. A claim is generally associated with a number of posts that are relevant to the claim. We are not interested at the individual level, but at the aggregate level. Therefore, predicting the veracity of each post is not our focus here. Instead, we concentrate on detecting rumors at the event-level, comprised of a set of relevant posts.

We define a set of given events as $E = \{E_i\}$, where each event $E_i = \{(m_{i,j}, t_{i,j})\}$ consists of ideally all relevant posts $m_{i,j}$ at timestamp $t_{i,j}$, and the task is to classify each event as a rumor or not.

4.2 Variable-length Time Series

We could model each post as an input instance and construct an RNN modeling the time series with a sequence length equal to the number of posts. However, there could be tens of thousands of posts in a popular event. We have only a single output unit indicating the class at the last time step of each event. Back propagation through a large number of time steps with only a final-stage loss will be computationally expensive as well as ineffective. Hence, we batch posts into time intervals and treat them as a single unit in a time series that is then modeled using an RNN sequence. A reference length of RNN sequence is adopted for constructing the time series.

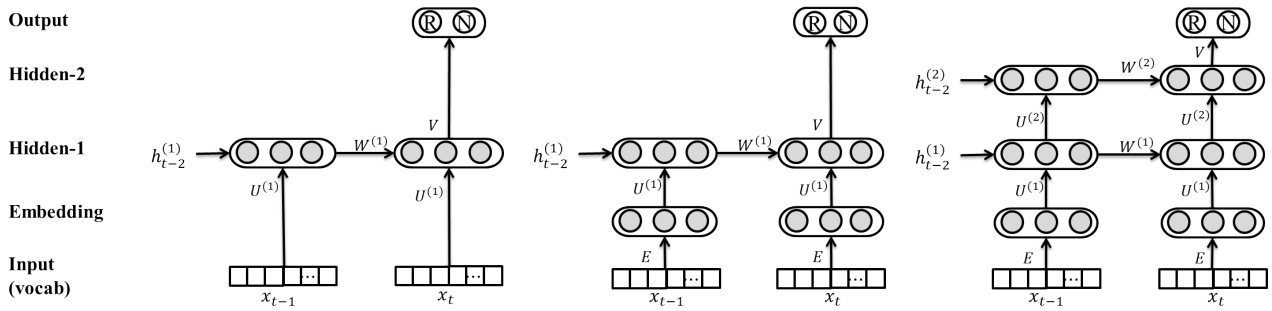
批量

Time spans representing densely populated with posts in the diffusion should be captured properly; the number of time intervals adopted *approximates* the reference length of RNN. Algorithm 1 describes the procedure. Initially, we divide the entire timeline equally into N intervals (i.e., N is the reference length). Then, our system tries to discover the set of non-empty intervals U' (i.e., each interval in U' has at least one tweet) by removing the empty ones in the set U_0, from which those continuous intervals whose overall time span is the longest are chosen into the set $\bar{U}$. If the number of intervals in $\bar{U}$ is lower than N and the number of intervals is more than that of the previous round, we halve the intervals and continue partitioning; otherwise, it returns the discovered continuous intervals given by $\bar{U}$. Note that the length of entire time series, though is close to N, varies among different events, whereas the length of individual intervals in an event is equal.

Input : Relevant posts of $E_i = \{(m_{i,j}, t_{i,j})\}_{j=1}^{n_i}$, Reference length of RNN N
Output: Time intervals $I = \{I_1, I_2, \ldots\}$

```
   /* Initialization                                   */
1  L(i) = t_{i,n_i} − t_{i,1};  ℓ = L(i)/N;  k = 0;
2  while true do
3      k ++;
4      U_k ← Equipartition(L(i), ℓ);
5      U_0 ← {empty intervals} ⊆ U_k;
6      U'_k ← U_k − U_0;
7      Find Ū_k ⊆ U'_k such that Ū_k contains continuous
         intervals that cover the longest time span;
8      if |Ū_k| < N && |Ū_k| > |Ū_{k−1}| then
           /* Shorten the intervals                    */
9          ℓ = 0.5 · ℓ;
10     else
           /* Generate output                          */
11         I = {I_o ∈ Ū_k | I_1, ..., I_{|Ū_k|}};
12         return I;
13     end
14 end
15 return I;
```

Algorithm 1: Algorithm for constructing variable-length time series given the set of relevant posts of an event and the reference length of RNN



(a) Basic tanh-RNN (b) 1-layer LSTM/GRU + embedding (c) 2-layer GRU + embedding

Figure 2: Our RNN-based rumor detection models. E is the word embedding weight matrix, U, W, V correspond to the parameters of hidden layers and output layers. R means rumor and N means non-rumor.

4.3 Structures of Models

Based on the times series constructed in Section 4.2, the recurrent units of RNN naturally fit the time intervals. In each interval, we use the $tf * idf$ values of the vocabulary terms in the interval as input. We prune the vocabulary by keeping the top-K terms according to their $tf * idf$ value, so the input dimension is K. We develop RNNs of three different structures as shown in Figure 2. Note that the output unit is associated with the last time step, which uses softmax for the probabilistic output of the two classes.

tanh-RNN is the fundamental structure, where the hidden units are not gated. Therefore, it may capture the context across the time intervals in a limited way.

Let g_c, where c denotes the class label, be the ground-truth 2-dimensional multinomial distribution of an event. Here, the distribution is of the form $[1, 0]$ for rumors and $[0, 1]$ for non-rumors. For each training instance (i.e., each event), our goal is to minimize the squared error between the probability distributions of the prediction and ground truth:

$$min \quad \sum_c (g_c - p_c)^2 + \sum_i ||\theta_i||^2$$

where g_c and p_c are the gold and predicted distributions, respectively, θ_i represents the model parameters to be estimated, and the L2-regularization penalty is used for trading off the error and the scale of the problem. The tanh-RNN is shown in Figure 2(a).

Single-layer LSTM and GRU. Long-distance dependencies are important for capturing the patterns in rumors and hidden signals over the life cycle of the event. We modify the recurrent unit into the gated units using LSTMs and GRUs. The gated-unit RNNs are shown in Figure 2(b). The gated units not only keep the content of the current time step but also inject the inter-dependent evidence from its previous steps.

However, the scale of parameters is significantly enlarged because of the gated units. For instance, GRUs triple the original parameter space due to the introduction of reset gates and update gates. To reduce the complexity, we add an embedding layer (with a fixed length of 100) between the input and hidden layers so that the overall scale of parameters becomes much smaller. Including the embedding layer converts the sparse input word vectors into low-dimensional representations. Instead of using pre-trained vectors based on external collections, we learn the embedding matrix E ourselves with our model.

Multi-layer GRU. We further develop a multiple layer structure based on GRUs by adding a second GRU layer that captures higher-level feature interactions between different time steps. Considering the more complex parameter set of LSTMs, we only extend GRUs into multiple layers. Figure 2(c) illustrates its structure. The state values of hidden units are calculated using equations 2-4, where the embedding layer is given as Equation 2, and the first and second GRU layers are given as Equations 3 and Equations 4, respectively.

$$x_e = x_t E \tag{2}$$

$$\begin{aligned} z_t^{(1)} &= \sigma\left(x_e U_z^{(1)} + h_{t-1}^{(1)} W_z^{(1)}\right) \\ r_t^{(1)} &= \sigma\left(x_e U_r^{(1)} + h_{t-1}^{(1)} W_r^{(1)}\right) \\ \tilde{h}_t^{(1)} &= tanh\left(x_e U_h^{(1)} + (h_{t-1}^{(1)} \cdot r_t^{(1)}) W_h^{(1)}\right) \\ h_t^{(1)} &= (1 - z_t^{(1)}) \cdot h_{t-1}^{(1)} + z_t^{(1)} \cdot \tilde{h}_t^{(1)} \end{aligned} \tag{3}$$

$$\begin{aligned} z_t^{(2)} &= \sigma\left(h_t^{(1)} U_z^{(2)} + h_{t-1}^{(2)} W_z^{(2)}\right) \\ r_t^{(2)} &= \sigma\left(h_t^{(1)} U_r^{(2)} + h_{t-1}^{(2)} W_r^{(2)}\right) \\ \tilde{h}_t^{(2)} &= tanh\left(h_t^{(1)} U_h^{(2)} + (h_{t-1}^{(2)} \cdot r_t^{(2)}) W_h^{(2)}\right) \\ h_t^{(2)} &= (1 - z_t^{(2)}) \cdot h_{t-1}^{(2)} + z_t^{(2)} \cdot \tilde{h}_t^{(2)} \end{aligned} \tag{4}$$

where E is the word embedding weight matrix, $(U^{(1)}, W^{(1)})$ and $(U^{(2)}, W^{(2)})$ are the weight connections inside the GRU units for the first and the second hidden layers, respectively.

Model Training. We train all the RNN models by employing the derivative of the loss through back-propagation [Collobert *et al.*, 2011] with respect to all of the parameters. We use the AdaGrad algorithm [Duchi *et al.*, 2011] for parameter update. We empirically set the vocabulary size K as 5,000, the embedding size as 100, the size of the hidden units as 100 and the learning rate as 0.5. We iterate over all the training

Table 1: Statistics of the datasets

Statistic	Twitter	Weibo
Users #	491,229	2,746,818
Posts #	1,101,985	3,805,656
Events #	992	4,664
Rumors #	498	2,313
Non-Rumors #	494	2,351
Avg. time length / event	1,582.6 Hours	2,460.7 Hours
Avg. # of posts / event	1,111	816
Max # of posts / event	62,827	59,318
Min # of posts / event	10	10

events in each epoch and continue until the loss value converges or the maximum epoch number is met.

5 Experiments and Results

5.1 Data Collection

We construct two microblog datasets using Twitter (`www.twitter.com`) and Sina Weibo (`weibo.com`). For the Twitter data, we confirmed rumors and non-rumors from `www.snopes.com`, an online rumor debunking service. We obtain 778 reported events during March-December 2015, of which 64% are rumors. For each event, we extract the keywords from the last part of the Snopes URL, e.g., http://www.snopes.com/**pentagon-spends-powerball-tickets**. We refine the keywords by adding, deleting or replacing words manually, and iteratively until the composed queries can have reasonably precise Twitter search results. We use scripts to download the "Live" search results from Twitter. To balance the two classes, we further added some non-rumor events from two public datasets [Castillo *et al.*, 2011; Kwon *et al.*, 2013]. The resulting dataset contains 498 rumors and 494 non-rumors.

For Weibo data, we obtain a set of known rumors from the Sina community management center[3], which reports various misinformation. The Weibo API can capture the original messages and all their repost/reply messages given an event. We also gather a similar number of non-rumor events 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. Table 1 provides more details.

5.2 Experimental Settings and Results

We compare the following approaches with our models:

SVM-TS: Ma, et al., [Ma *et al.*, 2015] proposed the linear SVM classifier that uses time-series structures to model the variation of social context features. We replicate their hand-crafted features based on contents, users and propagation patterns.

DT-Rank: Zhao, Resnick, and Mei proposed a decision-tree-based ranking model to identify trending rumors [Zhao *et al.*, 2015]. They search for enquiry phrases and cluster disputed factual claims, and rank the clustered results based on statistical features. We implement their enquiry phrases and features.

DTC and *SVM-RBF:* The Twitter information credibility model using Decision Tree Classifier [Castillo *et al.*, 2011] and the SVM-based model with RBF kernel [Yang *et al.*, 2012], both using hand-crafted features based on the overall statistics of the posts, rather than temporal information.

RFC: Kwon, et al., [Kwon *et al.*, 2013] proposed a Random Forest Classifier using three parameters to fit the temporal tweets volume curve. We extend it to fit the curves of the same set of hand-crafted features used by SVM-TS.

Our models: (1) *tanh-RNNs* is the basic RNN structure with a single hidden layer. (2) *LSTM-1* and *GRU-1* are configured with one-layer of LSTM and GRU hidden units, respectively, each with an embedding layer on the top of the input layer. (3) *GRU-2* is enhanced with an extra GRU hidden layer for capturing higher-level feature interactions.

We implement DTC and RFC using Weka[4], SVM models using LibSVM[5] and RNN models with Theano[6]. We hold out 10% of the events in each dataset for model tuning, and the rest of the events are split with a ratio of 3:1 for training and test. We use accuracy, precision, recall and F-measure as evaluation metrics. N is empirically set to 50.

Table 2 shows the performance of all the systems. Our models outperform all the baselines on both datasets. The simplest RNN model, tanh-RNN, achieves 82.7% accuracy on Twitter and 87.3% on Weibo. This result indicates that the basic RNN can learn discriminative features effectively. All the baselines resort to hand-crafted features or rules; hence, they are limited with respect to learning deep latent features and their correlations.

SVM-TS and RFC outperform other baselines because of the temporal structure they use, but they are still obviously worse than RNN-based models. The DT-Rank method uses a set of regular expressions. Only 1.63% Weibo posts and 10.9% tweets in our datasets contain these expressions. That is why the results of DT-Rank are not satisfactory.

For the four RNN-based models, the superiority of the gated units (LSTM and GRU) over the tanh unit is clear. This result implies that the gated units can capture the long-distance dependencies among the signals, which may reside in any time step. GRU-1 and LSTM-1 perform well; GRU-1 is slightly better.

The extent of improvement made by GRU-2 over GRU-1 is different on two datasets. The Twitter data is much more noisy than its Weibo counterpart. The extra hidden layer can help overcome the noise by capturing higher-level interactions more accurately. This observation also explains the obviously lower baseline accuracies on Twitter than on Weibo. The performance of the RNN models varies just a little across the two datasets.

Figure 3 shows the learning curves of RNN-based methods. In both datasets, the GRU and LSTM converge faster with lower loss than tanh-RNN.

[3] `http://service.account.weibo.com`

[4] `http://www.cs.waikato.ac.nz/ml/weka/`

[5] `https://www.csie.ntu.edu.tw/~cjlin/libsvm/`

[6] `http://deeplearning.net/software/theano/`

Table 2: Rumor detection results (R: Rumor; N: Non-rumor)

(a) Twitter dataset

Method	Class	Accuracy	Precision	Recall	F_1
DT-Rank	R	0.644	0.638	0.675	0.656
	N		0.652	0.613	0.632
SVM-RBF	R	0.722	0.856	0.526	0.651
	N		0.663	**0.914**	0.769
DTC	R	0.731	0.724	0.757	0.740
	N		0.739	0.704	0.721
RFC	R	0.772	0.717	0.908	0.801
	N		0.870	0.634	0.734
SVM-TS	R	0.808	0.735	**0.963**	0.834
	N		**0.947**	0.652	0.772
tanh-RNN	R	0.827	0.847	0.833	0.840
	N		0.804	0.820	0.812
LSTM-1	R	0.855	0.855	0.883	0.869
	N		0.854	0.820	0.837
GRU-1	R	0.864	**0.857**	0.900	0.878
	N		0.872	0.820	0.845
GRU-2	R	**0.881**	0.851	0.950	**0.898**
	N		0.930	0.800	**0.860**

(b) Weibo dataset

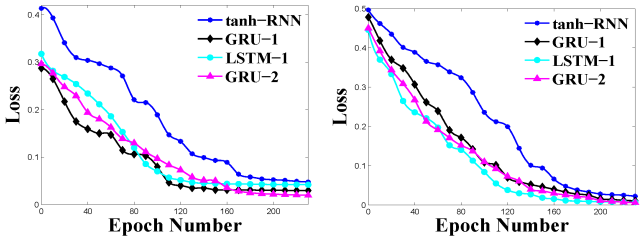
Method	Class	Accuracy	Precision	Recall	F_1
DT-Rank	R	0.732	0.738	0.715	0.726
	N		0.726	0.749	0.737
SVM-RBF	R	0.818	0.822	0.812	0.817
	N		0.815	0.824	0.819
DTC	R	0.831	0.847	0.815	0.831
	N		0.815	0.847	0.830
RFC	R	0.849	0.786	0.959	0.864
	N		0.947	0.739	0.830
SVM-TS	R	0.857	0.839	0.885	0.861
	N		0.878	0.830	0.857
tanh-RNN	R	0.873	0.816	0.964	0.884
	N		**0.956**	0.782	0.861
LSTM-1	R	0.896	0.846	**0.968**	0.913
	N		0.953	0.858	0.903
GRU-1	R	0.908	0.871	0.958	0.913
	N		0.953	0.858	0.903
GRU-2	R	**0.910**	**0.876**	0.956	**0.914**
	N		0.952	**0.864**	**0.906**

5.3 Early Detection Performance

For early detection, we make as a deadline the delay from the initial broadcast. Given a detection deadline, all the posts after it are invisible during the test. We use the mean official report time of rumors as a reference, i.e., the average report time over the rumors given by the debunking services of Snopes and Sina community management center.

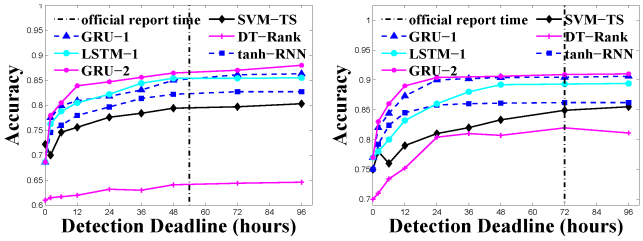
Figure 4 shows the accuracy of our models versus SVM-TS (the best baseline in Table 2) and DT-Rank (an early-detection-specific algorithm) for various deadlines. In the first few hours, the accuracy of the RNN-based methods climbs more rapidly and stabilize more quickly, indicating superior early detection performance of our method. Particularly, GRU-2 can detect rumors with 83.9% accuracy for Twitter and 89.0% for Weibo within 12 hours, which is much earlier than the baselines and the mean official report times.

Further analysis shows that events correctly detected as rumors by RNN demonstrate complex signals indicative of rumors. Table 3 gives a detected rumor about "Obama compiling a secret racial database", in which many questioning and denial signals (underlined) can be observed in the first few hours. Such indicators could be learned by the RNNs, while they are difficult to be accurately hand-crafted beforehand.



(a) Twitter dataset (b) Weibo dataset

Figure 3: Learning curves of RNN-based models



(a) Twitter dataset (b) Weibo dataset

Figure 4: Results of rumor early detection

Table 3: An example of the early detected rumor

1st hr	21:21:20	false messiah shepard justice plain wrong joke no fear
	21:27:48	what a wicked shocker! who ever would have guessed?
	21:30:17	terrifying, if true as if
	21:41:22	wtf?! mt
	21:43:56	what fresh hell
	21:45:08	what do you think about this
2nd hr	22:00:07	if this is true, he is coming for you and me
	22:04:49	truely evil!
	23:16:20	do you really believe honestly? truly?
	22:16:04	wow. a must read. if this is true, god help us
	22:35:38	another unbelievable hoping is not true
	22:52:10	wtf? new york post
3rd hr	23:06:07	imagine "wrong" hands not to like
	23:09:11	do you have to see christians to believe this?
	23:55:30	what could possibly go wrong?
	00:26:06	do you honestly believe?
	00:34:05	you hope this isn't true.

6 Conclusion

Most existing work on rumor detection from social media focus on extracting features or rules manually. In this research, we propose a deep learning framework for rumor debunking. Our method learns RNN models by utilizing the variation of aggregated information across different time intervals related to each event. We empirically evaluate our RNN-based method with three widely used recurrent units, tanh, LSTM and GRU, which perform significantly better than the state-of-the-arts. For further improvement, we add multiple hidden layers and embedding layers. There is still room to improve the method. To understand better how deep learning helps the rumor detection, more thorough experiments will be required in the future. In addition, we can also develop unsupervised models due to massive unlabeled data from social media.

Acknowledgement

Kam-Fai Wong was concurrently affiliated with MoE Key Laboratory on High Confidence Software Technologies (CUHK Sub-Lab), China and partly support by CUHK direct grant (4055055). Meeyoung Cha was partly supported by the National Research Foundation of Korea (NRF-2015K1A3A1A16002183)

References

[Allport and Postman, 1965] G.W. Allport and L.J. Postman. *The psychology of rumor*. Russell & Russell, 1965.

[Bengio *et al.*, 1994] Yoshua Bengio, Patrice Simard, and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 5(2):157–166, 1994.

[Castillo *et al.*, 2011] Carlos Castillo, Marcelo Mendoza, and Barbara Poblete. Information credibility on twitter. In *Proceedings of WWW*, 2011.

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

[Collobert *et al.*, 2011] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12:2493–2537, 2011.

[Devlin *et al.*, 2014] Jacob Devlin, Rabih Zbib, Zhongqiang Huang, Thomas Lamar, Richard Schwartz, and John Makhoul. Fast and robust neural network joint models for statistical machine translation. In *Proceedings of ACL*, 2014.

[Duchi *et al.*, 2011] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12:2121–2159, 2011.

[Friggeri *et al.*, 2014] Adrien Friggeri, Lada A Adamic, Dean Eckles, and Justin Cheng. Rumor cascades. In *Proceedings of ICWSM*, 2014.

[Graves, 2013] Alex Graves. Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850*, 2013.

[Gupta *et al.*, 2013] Aditi Gupta, Hemank Lamba, Ponnurangam Kumaraguru, and Anupam Joshi. Faking sandy: characterizing and identifying fake images on twitter during hurricane sandy. In *Proceedings of WWW companion*, 2013.

[Hochreiter and Schmidhuber, 1997] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[Hu *et al.*, 2013] Xia Hu, Jiliang Tang, Yanchao Zhang, and Huan Liu. Social spammer detection in microblogging. In *Proceedings of IJCAI*, 2013.

[Kombrink *et al.*, 2011] Stefan Kombrink, Tomas Mikolov, Martin Karafiát, and Lukás Burget. Recurrent neural network based language modeling in meeting recognition. In *Proceedings of INTERSPEECH*, 2011.

[Kwon *et al.*, 2013] Sejeong Kwon, Meeyoung Cha, Kyomin Jung, Wei Chen, and Yajun Wang. Prominent features of rumor propagation in online social media. In *Proceedings of ICDM*, 2013.

[LeCun *et al.*, 2015] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[Lee *et al.*, 2013] Kyumin Lee, James Caverlee, and Steve Webb. Uncovering social spammers: Social honeypots + machine learning. In *Proceedings of SIGIR*, 2013.

[Liu *et al.*, 2015] Xiaomo Liu, Armineh Nourbakhsh, Quanzhi Li, Rui Fang, and Sameena Shah. Real-time rumor debunking on twitter. In *Proceedings of CIKM*, 2015.

[Ma *et al.*, 2015] Jing Ma, Wei Gao, Zhongyu Wei, Yueming Lu, and Kam-Fai Wong. Detect rumors using time series of social context information on microblogging websites. In *Proceedings of CIKM*, 2015.

[Ratkiewicz *et al.*, 2011a] Jacob Ratkiewicz, Michael Conover, Mark Meiss, Bruno Gonçalves, Alessandro Flammini, and Filippo Menczer. Detecting and tracking political abuse in social media. In *Proceedings of ICWSM*, 2011.

[Ratkiewicz *et al.*, 2011b] Jacob Ratkiewicz, Michael Conover, Mark Meiss, Bruno Gonçalves, Snehal Patil, Alessandro Flammini, and Filippo Menczer. Truthy: mapping the spread of astroturf in microblog streams. In *Proceedings of WWW*, 2011.

[Rumelhart *et al.*, 1986] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *Nature*, 323:533–536, 1986.

[Sutskever *et al.*, 2011] Ilya Sutskever, James Martens, and Geoffrey E Hinton. Generating text with recurrent neural networks. In *Proceedings of ICML*, 2011.

[Sutskever *et al.*, 2014] Ilya Sutskever, Oriol Vinyals, and Quoc VV Le. Sequence to sequence learning with neural networks. In *Proceedings of NIPS*, 2014.

[Wang, 2010] Alex Hai Wang. Don't follow me - spam detection in twitter. In *Proceedings of SECRYPT*, 2010.

[Wu *et al.*, 2015] Ke Wu, Song Yang, and Kenny Q Zhu. False rumors detection on sina weibo by propagation structures. In *Proceedings of ICDE*, 2015.

[Yang *et al.*, 2012] Fan Yang, Yang Liu, Xiaohui Yu, and Min Yang. Automatic detection of rumor on sina weibo. In *Proceedings of the ACM SIGKDD Workshop on Mining Data Semantics*, 2012.

[Zhao *et al.*, 2015] Zhe Zhao, Paul Resnick, and Qiaozhu Mei. Enquiring minds: Early detection of rumors in social media from enquiry posts. In *Proceedings of WWW*, 2015.