Retweet Prediction with Attention-based Deep Neural Network

Qi Zhang, Yeyun Gong, Jindou Wu, Haoran Huang, Xuanjing Huang
Shanghai Key Laboratory of Data Science
School of Computer Science, Fudan University
825 Zhangheng Road, Shanghai, P.R.China
{qz, yygong12, jdwu15, huanghr15, xjhuang}@fudan.edu.cn

ABSTRACT

On Twitter-like social media sites, the re-posting statuses or tweets of other users are usually considered to be the key mechanism for spreading information. How to predict whether a tweet will be retweeted by a user has received increasing attention in recent years. Previous methods studied the problem using various linguistic features, personal information of users, and many other manually constructed features to achieve the task. Usually, feature engineering is a laborious task, we require to obtain the external sources and they are difficult or not always available. Recently, deep learning methods have been used in the industry and research community for their ability to learn optimal features automatically and in many tasks, deep learning methods can achieve state-of-the art performance, such as natural language processing, computer vision, image classification and so on. In this work, we proposed a novel attention-based deep neural network to incorporate contextual and social information for this task. We used embeddings to represent the user, the user's attention interests, the author and tweet respectively. To train and evaluate the proposed methods, we also constructed a large dataset collected from Twitter. Experimental results showed that the proposed method could achieve better results than the previous state-of-the-art methods.

Keywords

Retweet Prediction; Attention Mechanism; Deep Neural Network

1. INTRODUCTION

Because of their easy real-time information sharing capability, social media sites (e.g., Twitter, Facebook, and YouTube) have rapidly increased in recent years. According to the statics for Twitter, there are more than one billion unique visits monthly [1]. Users post a tweet on Twitter include a time stamp which reflects the time of tweet was posted and the author's unique identifier. A tweet is limited to 140 characters. Users can post a tweet, retweet a tweet or respond to a tweet. Retweets mean of taking part in the diffuse conversation. When users find interesting tweets or statuses and want to share them with their followers, they can retweet them using a button or other mechanism to copy and post these tweets. The information could spread on the network remains relatively intact of the original author. The retweet function provided by these services is usually considered to be the key mechanism for spreading information among users. Figure 1 shows some examples of the retweet behavior. Hence, the behaviors in social networks have been carefully studied [7, 46, 2, 52]. In addition to analyzing the retweeting behavior itself, retweeting can also help with a variety of tasks such as information spreading prediction [50, 34, 35, 48], popularity prediction [19], and tweet retrieval [26].

Figure 1: Examples of Retweet Behavior.

Due to the enormous usefulness of prediction, a variety of studies have been conducted on the task of automatically predicting in social network [49, 33, 27, 8, 54, 11, 12]. These studies investigated the problem from different points of view. Yang et al. [49] defined the task of predicting whether a user will retweet a tweet to their friends after viewing it. A factor graph model [49], conditional random fields [33], a topical model [54], and a ranking based method [27] have all been evaluated. Because this task is related to users and tweets, most of the previous studies took both the contextual and social information into consideration. However, most of these used manually designed and constructed features to achieve the task.

In this study, we investigated the problem of predicting retweet

[1]https://about.twitter.com/company

CIKM'16 , October 24-28, 2016, Indianapolis, IN, USA

DOI: http://dx.doi.org/10.1145/2983323.2983809

behavior. This task has several challenges from different perspectives. From the linguistic point of view, in Twitter-like services, users usually write tweets using a conversational style. Tweets are known to be noisy and contain various oral expressions. The language species used in Twitter also contain various types. The performance of methods based on traditional manually constructed features may sharply change for different kinds of datasets. Another challenge is that the task is related not only to the tweet itself but also to its author and the user we need to predict. Zhang et al. [54] also mentioned that whether a user will retweet a tweet or not is based on the following three main factors: 1) the author information, 2) content information, and 3) user interests. Hence, the retweet prediction method should take all of these information into consideration. Moreover, a large-scale dataset that contains contextual and social information is greatly needed to train and evaluate the methods. All of these factors make it a challenging task. Additionally, most of the existing methods only consider the surface information of the tweet and ignore the attention interests of users or similarity information between the tweet and user interests.

To meet these challenges, in this work, we proposed a novel attention-based deep neural network to achieve this task. The proposed method combines the content of the tweet, the user interests, the similarity information between the tweet and user interests, user information and author information. All these factors are first converted to the representation of embeddings. Because users may post a variety of tweets, and these tweets have different priorities, in this model, we also design an attention mechanism to encode the interests of the user. The retweet behavior is predicted through a fully connected softmax function.

The main contributions of this work can be summarized as follows:

- We propose a novel attention-based deep neural network to incorporate the user, author, user interests, and similarity information between the tweet and user interests to predict the retweet behavior.
- We construct a large collection of tweets from Twitter, which contains both the tweet content and social network information of related users.
- Experimental results demonstrate that the proposed method can achieve better performance than state-of-the-art methods.

2. RELATED WORK

2.1 Convolutional Neural Network

Convolutional Neural Network has been successfully applied to many fields, such as Natural Language processing, Image recognition or Video processing and so on. [5, 23] used a convolutional architecture for the sentences modelling. [22] study multiple approaches for extending the connectivity of a CNN in time domain for the videos classification. [44] combine the representational power of large, multilayer neural networks together for scene text recognition. [24] used a deep convolutional architecture for the image classification. [1] explore the convolutional neural network(CNN) in multiple dimensions, they study different convolutional architectures and proposed a softmax pooling layer with weight which can automatic learn the pooling size of the speech recognition task. [32] design a method to train layers on the ImageNet dataset and reuse the layers to compute mid-level image representation for other image dataset. [20] use the convolutional neural network to do the sentences matching task through adapting the convolutional architecture in vision and speech. In their model, they denote the hierarchical structures of sentences using a layer-by-layer composition and pooling, and they capture the matching patterns in their model from different levels. [39] proposed a method to learn low-dimensional semantic vectors using a series of latent semantic models which are based on a convolutional neural network for search queries and Web documents. [21] proposed a 3D model based on CNN to do action recognition. This model extracts features from both the temporal dimensions and the spatial through using 3D convolutions. [44] proposed a method to combine unsupervised feature learning with multi-layer neural networks for the representational power to do the full end-to-end text recognition task in natural images problem. [13] proposed a method to represent the meaning of documents through embedding them into a low dimensional vector space. In their method, they use convolutional neural network to embed the documents and preserve the order of word and sentence to capture nuanced semantics. [37] proposed a convolutional architecture to rerank pairs of short texts, their model can learn a optimal representation of text pairs and relate them in a supervised way by a similarity function. [36] proposed a recurrent convolutional neural network to consider a large input context for scene parsing. Their method does not rely on task-specific features and segmentation methods, the method is trained using an end-to-end mechanism over raw pixels. [16] proposed a deep semantic similarity model, their method design a special type of deep neural networks for text analysis, and based on the source document of the user is reading, it can recommend interest target documents to the user.

2.2 Attention-based Neural Network

Attention-based neural network has been used in various tasks and achieves promising performance recently, such as machine translation [28], speech recognition [3, 10], visual object classification [31] and so on. [28] proposed an attention-based long short term memory combine the global attention and local attention for the machine translation task. [38] introduced a soft attention method which combine the convolutional neural network and long short term memory for the action recognition. [31] introduced an attention-based recurrent neural network to select a sequence of locations or regions for the further processing. [3] introduced an attention recurrent neural network for the speech recognition task. This approach replace the HMM by a Recurrent Neural Network(RNN) which perform sequence prediction at the character level. [10] modify the attention mechanism which avoid to concentrate the attention on a single frame, and adding location awareness to the attention mechanism, they proposed a generic method for the speech recognition. [9] proposed the correct answers to a question requiring the model's attention focus on the corresponding regions. And they used an attention based convolutional neural network to learn the corresponding regions to the question for visual question answering task. [17] proposed a Deep Recurrent Attentive Writer neural network architecture to combine a novel spatial attention mechanism with a sequential variational auto-encoding framework for image generation. [47] proposed an attention based model for the task to learn the description of the content for images automatically. [45] integrates three types of attention: the bottom-up attention, the object-level top-down attention, and the part-level top-down attention to apply visual attention to fine-grained classification task based on deep neural network.

2.3 Retweeting

Retweeting is the main way to diffuse information from twitter,

so many researchers focused on this task recently. [42] investigate how political discussions take place in the Twitter network during periods of political elections with a focus on the most active and most influential users. [7] maps out retweeting as a conversational practice. [43] examine a number of features that might affect retweetability of tweets. [15] study how to learn a predictive model to rank the tweets according to their probability of being retweeted. [54] proposed a method using non-parametric statistical models to combine structural, textual, and temporal information together to predict retweet behavior. [4] proposed two Bayesian nonparametric models on retweet data to integrate users' retweet behavior and the analysis of tweet text in the same probabilistic framework. [27] explore the features: followers status, retweet history, followers interests and followers active time with a learning to-rank framework for finding who will retweet a tweet posted on Twitter. [15] regard the users, publishers and tweets as three types of nodes to build a graph to learn a predictive model for ranking the tweets according to the probability of being retweeted. [8] proposed a method to use visual cues of an image which is linked in a tweet, content of the tweet and structure-based features to predict the expected retweet count of the tweet. [53] proposed two instantiation functions based on structural diversity and pairwise influence and introduce a notion of social influence locality to predict users' retweet behaviors. [25] used the PageRank method on the retweet graph to measure of influence of users, and combined with the flow of the cascade as the new features to predict the times of retweets during epoch T. [41] used the message itself, the external context, the features about the users involved and the relationship between the time to build a preliminary model for predicting the time between information redistribution and dissemination on Twitter.

3. PROPOSED METHOD

In this work, we formulate the task of retweet prediction as a binary classification problem. Given a tweet t and user u, our task is to classify it as a positive or negative. We use neural network to handle input tweets of varying lengths. The output layer represents the probability of retweeting. In our model, we incorporate the user, author, user interests, the content of the tweet and the similarity between user interests and tweet with an attention mechanism neural network for the retweet prediction. Figure 2 shows the architecture of the proposed model.

We use convolutional neural network for the content of the tweet encoding. And we use an attention-based neural network to encode the attention interests of the user. After we encode the user interests and the content of the tweet, we compute the similarity score with a similarity matrix for the user's attention interests and tweet. To encode the user and author consistencies, we encode each user and author with continuous vectors $v_u \in R^{d_u}$ and $v_a \in R^{d_a}$ respectively, where d_u is the dimension of the user vector, and d_a is the dimension of the author vector. Then, we employ a concatenation layer to combine the information from all the vectors to produce a hidden state as follows:

$$\widehat{\mathbf{h}} = \mathbf{v}[\mathbf{v}_u; \mathbf{v}_i; s; \mathbf{v}_p; \mathbf{v}_a], \tag{1}$$

where $\mathbf{v}_i$ is the embedding of the user's attention interests. $\mathbf{v}_p$ is the feature vector of the tweet for prediction, s is the similarity score between the user's attention interests and tweet.

Finally, we use a fully connected softmax function for the retweet prediction problem. The parameters in the models are learned jointly with our final objective function instead of being trained separately.

3.1 The Variants of Convolutional Neural Network

Before introducing our attention based deep neural network, we will introduce two variants of convolutional neural network: convolutional neural network with user information(U-CNN) and convolutional neural network with user and author information(UA-CNN).

Various sentence models can be used for the tweet encoding, such as the neural bag-of-words models, recurrent neural network[29], recursive neural network[40] and so on. In this work, we use the convolutional architecture to model the tweet, and then we combine the user and author information for the prediction as shown in Figure 3.

In the input layer, we need two steps to convert a tweet to the format of the input. A tweet is a sequence of words. First, we need convert each word in the tweet to a word vector. Second, we concatenate these word vectors to build the tweet matrix of input. In the matrix, each column is a feature vector corresponding to a word:

$$vs = \begin{bmatrix} | & | & | \\ v_1 & ... & v_n \\ | & | & | \end{bmatrix}. \tag{2}$$

We use the convolutional layer to extract local features for each word with its context. In the convolutional layer, we first generate a contextual vector by select a window size l and concatenate these word vectors in the window. Then, we using a filter matrix $\mathbf{w} \in R^{l \times d}$ and a non-linear function to operates on the contextual vector. The output for one operation of the filter matrix and non-linear function is a local feature. The following is an example of this operation:

$$vf_j = g(\mathbf{w} \cdot \begin{bmatrix} v_j \\ \vdots \\ v_{j+l-1} \end{bmatrix} + b), \tag{3}$$

where $\mathbf{w}$ is the filter matrix of the convolutional layer, g is a non-linear function, we used rectified linear unit(relu) proposed in [24] as the non-linear function. In this method, the neuron's output vf is a function of $f(x) = max(0, x)$. $b \in R$ is a bias term. We perform this operation on different combinations of continuous feature vectors $\{\mathbf{v}_{1:l}, \mathbf{v}_{2:l+1}, ..., \mathbf{v}_{n-l+1:n}\}$ of the words. Then we obtain a set of features from the filter.

$$\mathbf{vf} = [vf_1, vf_2, ..., vf_{n-l+1}], \tag{4}$$

where $\mathbf{vf} \in R^{n-l+1}$ is a feature vector with length of $n - l + 1$ and this feature vector is the input of the pooling layer.

The size of the local feature vectors extracted from the convolutional layer depends on the number of words in the tweet. While we need to combine the local feature vectors to generate a global feature vector, and the size of the local feature vectors should be fixed and independent of the tweet length. So, in the pooling layer, we extract the maximum value for each feature vector with a max-overtime pooling operation for each filter matrix as follows:

$$vf_{max} = \begin{bmatrix} max(vf_1) \\ \vdots \\ max(vf_n) \end{bmatrix}. \tag{5}$$

In this operation, the most important feature can be captured by extracting the highest value for each set of features. This pooling operation can be used for variable tweet lengths. And the output is a fixed size feature vector.

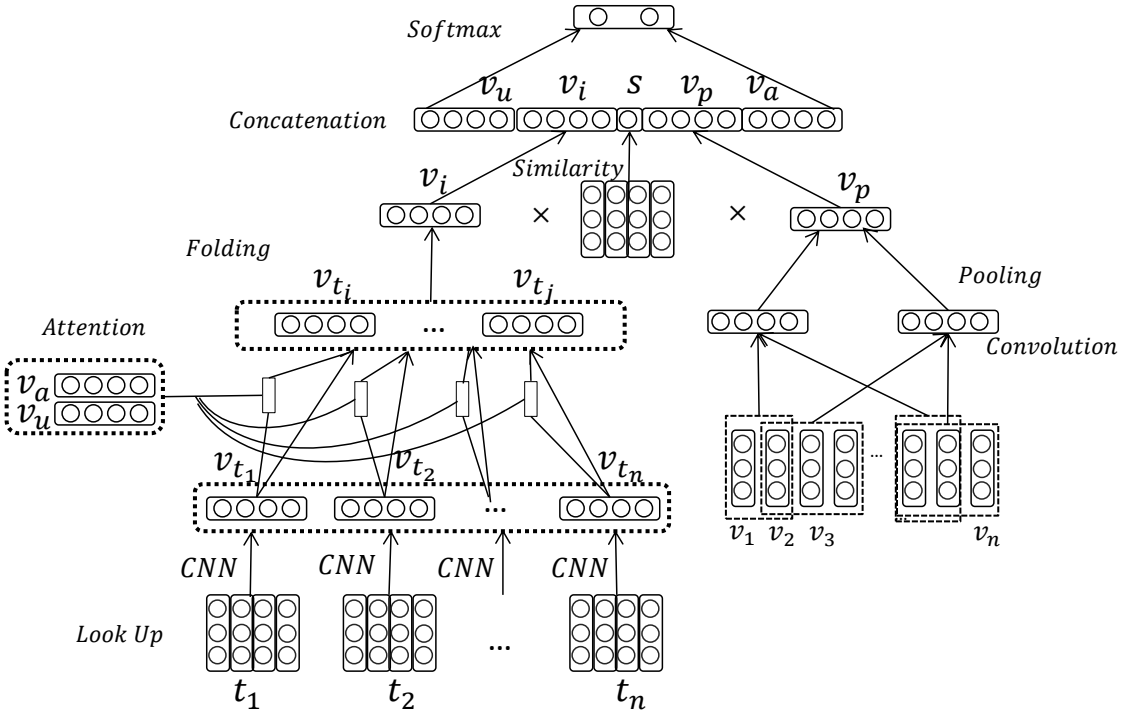


Figure 2: An illustration of the neural network approach for retweet prediction.

From the process described above, we can see that each filter can produce one feature. For each window size, we use different filters to produce more features. To capture various context information, we vary the window sizes and obtain multiple sizes of filters to produce features in the model. After the max-overtime pooling operation, a non-linear function $relu$ with a bias is applied to the pooled matrix.

The output of the convolutional neural network is a fixed-length vector, which represents the embedding of the input tweet m. We encode each user and author with continuous vectors $v_u \in R^{d_u}$ and $v_a \in R^{d_a}$. Then, we combine the output feature vector with user vector to produce the hidden layer in the model U-CNN, The graph model has been shown in Figure 3(a). While in the model UA-CNN, we employ a concatenation layer to combine the output feature vector with user vector and author vector to produce the hidden state as shown in Figure 3(b). Finally, we use a fully connected softmax function for the retweet prediction problem in both U-CNN and UA-CNN.

3.2 Attention-based Convolutional Neural Network

In the previous section 3.1, we introduce the convolutional neural network with user and author information. This method embed the tweet with convolutional neural network, and combines the embeddings of user, author and tweet with a hidden layer. However, the history tweets posted by the user which reflect the user's interests have not been concluded in this method. Therefore, in the attention based convolutional neural network(SUA-CNN), we combine the user's attention interests and the similarity score of the user's attention interests and tweet. Finally, we predict the retweet behavior based on the combined features. The model is shown in Figure 2.

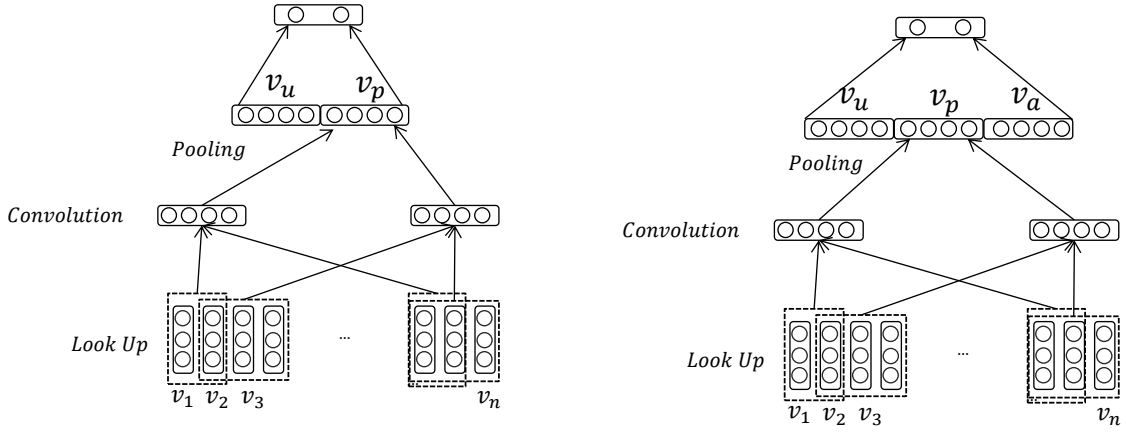
3.2.1 User Interests Encoding with Attention Neural Network

We embed the tweets of the user as the user's interests. Many users have more than one thousands tweets, while the tweets only focus on limited interests. Thus, we first cluster the tweets of the user into n clusters using K-means. We represent the tweets with $t_1, t_2, ..., t_m$. The K-means method will cluster the tweets into n groups and the algorithm is shown in Alg. 1. The tweets in each cluster represent one interest of the user. Then, we extract the central tweet for each cluster as one interest of the user. Through this step we extract n tweets of the user. Generally, each interest has a different importance weight for the retweet behavior of the user for the specific tweet. To extract the attention weight, we design an attention layer.

Given the input tweets T, we first encode each tweet using a simple convolutional neural network(CNN). The word embeddings input a convolutional layer with multiple window sizes, and then through the pooling layer, we capture the most important feature using a max-overtime pooling operation. Finally, we regard the output of the CNN as the embedding $\mathbf{v_t}$ of the tweet.

Algorithm 1 Clustering operation with K-means

Random select n cluster centroids:$\eta_1, \eta_2, ..., \eta_n$
while Convergence condition is not satisfied: **do**
  for each tweet t_i: **do**
    Compute it's cluster with:$c^i = \arg\min_j ||t_i - \eta_i||^2$
  end for
  for each cluster j: **do**
    Compute the cluster centroid with: $\eta_j = \frac{\sum_{i=1}^{m} 1\{c^i=j\} t_i}{\sum_{i=1}^{m} 1\{c^i=j\}}$
  end for
end while



(a) U-CNN (b) UA-CNN

Figure 3: The illustration of two variants of Convolutional Neural Network. (a) is the Convolutional Neural Network with User information, and (b) is the Convolutional Neural Network with User and Author information (UACNN).

After converting tweets into embeddings, the next step is the attention layer. We take the embeddings $\mathbf{v_{t_i}} \in R^k$ for each tweet with user vector $\mathbf{v}_u$ and the author embedding $\mathbf{v}_a$ to obtain the input of the attention layer. Where k is the dimension of the tweet vector. The attention layer combines the vectors of the user, the author, and one tweet of the user to generate the weight for the tweet of the user as follows:

$$q_i = \mathbf{w} * h[\mathbf{v}_{t_i}; \mathbf{v}_u; \mathbf{v}_a], \tag{6}$$

where $\mathbf{v}_{t_i}$ is the embedding of the tweet t_i. $\mathbf{v}_u$ is the embedding of the user u, $\mathbf{v}_a$ is the embedding of the author. $h[\mathbf{v}_{t_i}; \mathbf{v}_u; \mathbf{v}_a]$ is the concatenate of $\mathbf{v}_{t_i}$, $\mathbf{v}_u$ and $\mathbf{v}_a$. q_i is the important weight of the tweet t_i. Then we normalize the weights with:

$$s_i = \frac{exp(q_i)}{\sum_{i \in F} exp(q_j)} \quad i \in F \tag{7}$$

Where F is the weight set. The attention layer makes it possible to attention on the interests and for different interests operated with different weights.

After the attention layer, the embeddings of the extracted attention interests will be performed by the folding layer, which can operate on different numbers of tweets. The folding layer is used to abstract the features of the attention interests using the following operation.

$$f_i = g(\sum_j (s_j * \widehat{v}_{j,i})), \tag{8}$$

where $\widehat{\mathbf{v}}_{\mathbf{i,j}}$ is the value in the ith position of the embedding of the jth attention interest. The folding is the sum operation for each dimension of all the attention interests. g is a non-linear function.

The final output of this neural network is a fixed-length vector, which represents the embeddings of the attention interests $\mathbf{v}$.

3.2.2 The Similarity of Tweet and User Interests

The tweet model based on convolutional neural network as described in Section 3.1 used to map the target tweet into a vector, and the attention model described in Section 3.2.1 used to compute the attention weight for each user's interest and encoded to a vector. Then, we use the vector of the target tweet and the vector of the user's attention interests to compute their similarity score. Finally we joined the similarity score, the vector of the target tweet, the vector of the user's attention interests, the embedding of the user, and the embedding of the author in a single representation.

In the following, we introduce the method to compute the similarity score of the target tweet and user's attention interests. Then we will introduce the remaining layers: hidden layer and softmax layer.

Given the tweet embedding v_p from our convolutional neural network and the user's attention interests embedding v_i from our attention-based convolutional neural network. Then, we can compute the similarity score between v_p and v_i by the method proposed in [6]. This method defines the similarity score as follows:

$$sim(v_p, v_i) = v_p^T M v_i, \tag{9}$$

where $M \in R^{k \times k}$ is a similarity matrix. The method to compute score in the Eq. 10 is similar to the approach of the noisy channel in machine translation which has been used in question answering and information retrieval [14]. The parameter matrix M will be optimized in the training process.

In the hidden layer we combine the similarity score, the user embedding, the author embedding, the attention interests vector and tweet vector to a fixed-size feature vector. Then the vector is fed to a network layer with nonlinear activation function. Finally, we extract the highly non-linear features from the output layer as follows:

$$\mathbf{f_h} = \mathbf{g}(\mathbf{w_h} \cdot \mathbf{h}[\mathbf{v_u}, \mathbf{v_i}, \mathbf{s}, \mathbf{v_p}, \mathbf{v_a}] + \mathbf{b}), \tag{10}$$

where $\mathbf{w_h}$ is the parameter vector, b is the bias and g is a non-linear function.

The output of the hidden layer is the highly non-linear features which is fed to a fully connected softmax layer. From the softmax layer, we can predict the retweet probability by $p(y = i|\mathbf{v}) = \frac{exp(\theta_i \mathbf{v}^T)}{\sum_j exp(\theta_j \mathbf{v}^T)}$ where $mathbfv$ is the highly non-linear features vector of the raw input features through a series of operations from the attention-based deep neural network. θ_j is a parameter vector of the j-th label.

3.3 Regularization

We use the dropout [18] for the regularization. For the output of a hidden neuron, we set it to zero with probability p, and an output of zero will not contribute to the forward pass. In the backpropagation, the gradients are only backpropagated through the neurons without "drop out". In this way, the neural network has a different architecture each time an input is presented. However, these architectures share the weights. Consider the hidden layer h of the network. Let $\mathbf{x}$ represent the feature vector of input into this layer, L^p represents the feature vector of output from the pre-layer and L^c represents the feature vector of output from the current layer. We can compute the feed-forward operation for the original neural network as follows:

$$\begin{aligned} \mathbf{x} &= w \cdot L^p + b, \\ L^c &= g(x), \end{aligned} \tag{11}$$

where $\mathbf{w}$ is the weight vector and $\mathbf{b}$ is the bias. g is a non-linear activation function.

We can compute the feed-forward operation with dropout:

$$\begin{aligned} q &\sim Bernoulli(p), \\ L^p &= q * L^p, \\ x &= w \cdot L^p + b, \\ L^c &= g(x), \end{aligned} \tag{12}$$

where $*$ is an element-wise product. q is a vector of independent Bernoulli random variables each of which has probability p of being 1. This vector is sampled and multiplied element-wise with the outputs of that layer, y, to create the thinned outputs y.

In this method, each neuron relies on the different presence of other neurons, and more robust features can be learned. In the prediction process, the weight vectors learned will be multiplied by p. We also constrain l_2-norms of the weight vectors as follows:

$$||w||_2 = \eta, \quad if \ \ ||w||_2 > \eta. \tag{13}$$

This constraining will be performed whenever $||w||_2 > \eta$, after the gradient descent step.

3.4 Training

In this work, we learned the parameters Θ using a deep neural network.

$$\Theta = \{\mathbf{V}, \mathbf{M}, \mathbf{v}_u, \mathbf{v}_a, \mathbf{W}_u, \mathbf{W}_a\}, \tag{14}$$

here $\mathbf{V}$ are the words embeddings. $\mathbf{M}$ is the similarity matrix. $\mathbf{v}_u$ is the user vector. $\mathbf{v}_a$ is the author vector. $\mathbf{W}_u$ are the parameters in the user's interests encoding process, and $\mathbf{W}_a$ are the parameters in the tweet encoding process. The rest of the parameters belong to the fully connected layer. Our training objective function is formulated as follows:

$$J = \sum_{(m,i) \in D} -logp(i|m), \tag{15}$$

here D is the training corpus, $i \in \{0, 1\}$ is the user retweet behavior of tweet m, when $i = 1$ represents the user will retweet this tweet, $i = 0$ denotes will not retweet it.

To minimize the objective function, we use stochastic gradient descent(SGD) with the Adadelta update rule [51]. In the Adadelta update rule, the default learning rate is eliminated from the update rule, and not need to be set.

3.5 Retweet Prediction

We perform the retweet prediction as follows. Suppose that an unlabelled dataset is given. We first train our model using a training data, and save the model that has the best performance with the validation dataset. We encode the tweet of the unlabelled data using the saved model.

After the encoded processes, we combine the features generated from the process described in Section 3. Then, we predict the dth tweet in the unlabelled data using the fully connected layer:

$$P(y^d = i|\mathbf{h}^d, \mathbf{W}, \mathbf{b}) = \frac{exp(W_i \mathbf{h}^d)}{\sum_j exp(W_j \mathbf{h}^d)}, \tag{16}$$

where $\mathbf{W}$ are the parameters. $\mathbf{h}$ is the feature vector connected from the neural network. According to the scores output from fully connected layer, we can predict the retweet behavior of users.

4. EXPERIMENT

4.1 Data Construction

To analyze the retweet behavior, we crawled a large number of tweets based on the properties of the tweet service. We collected a data set from Twitter in the following ways. First, we randomly selected 5,000 users and crawled their tweets. In this step, we crawled 5,285,000 tweets. Among these users, we find that there are 2,985 users with retweet behavior, and the total number of tweets generated from retweet is 1,351,570. Second, we random selected 1,000 users and collected their retweets for evaluation. We assumed that the users could see the latest five tweets posted by their friends each time they retweeted. To restore what they have saw and ensure information integrity for the evaluation, we removed the retweeted tweets of the authors that had not been crawled from our data. Then, we collected the latest five tweets from their friends for each retweeted tweet. Finally, we removed the special characters and stopwords of all the tweets and dropped the tweets that have less than five words. After these steps, 37,615 tweets were left. We randomly selected 75% as training data, and the other 25% as test data.

Table 1 lists the statistics of the collected data set. From this table, we can observe that about 60% of the users have retweet behavior.

Table 1: Statistics of the data set

# Users	5,000
# Retweet Behavior Users	2,985
# Tweets	5,285,000
# Retweets	1,351,570

4.2 Experiment Configurations

We use the precision (P), recall (R), and F1-score (F_1) to evaluate the performance. For the convolutional filter, we used multiple window sizes (1,2), and the feature maps number is 100 for each filter. We set the dropout rate p to 0.5, and the l_2 constraint to 3. The mini-batch size of the training process is 40, and we set the cluster number to 5.

We randomly selected 10% of the training data as the dev set and performed early stopping on the dev set. In the training process, we used the robust stochastic gradient descent with the Adadelta update rule [51].

To initialize the word vectors, the publicly available word2vec vectors are used in this paper. They were trained using the continuous bag-of-words model proposed in [30] from Google News, which has 100 billion words. The dimension of the vectors is

Table 2: The performances of different methods in the test dataset.

Method	P	R	F_1
Random	0.279	0.502	0.358
Ave-SVM	0.363	0.624	0.459
Sum-SVM	0.380	0.654	0.481
CNN	0.558	0.306	0.395
ASC-HDP	0.709	0.611	0.656
U-CNN	0.655	0.523	0.582
UA-CNN	**0.789**	0.617	0.693
SUA-ACNN	0.733	**0.708**	**0.721**

300. For the words not in the vocabulary of pre-trained words, we initialized them with random vectors. To initialize the user vectors, and author vectors, we used random continuous vectors for each user and author with 300 dimensions.

For comparison with the proposed model, we also evaluated the following methods on the constructed dataset:

- **Random**: The retweet prediction is a binary classification task. For each tweet, we randomly select a decision of retweet or not.
- **Ave-SVM**: We regard the user and author as feature words and embedded them to vector respectively, then we averaged all the word vectors as the feature vector of the tweet. For the problem of positive and negative samples are not balanced, we use down-sampling to reduce the negative samples, then we obtain the positive and negative samples ratio of 1 to 1 in the training data. Finally we used these features to train a support vector machine classifier.
- **Sum-SVM**: In the method Sum-SVM, we sum all the word vectors as the feature vector of the tweet. And we also use the down-sampling method to sample the negative samples as the method Ave-SVM. Finally we used these features to train a classifier.
- **CNN**: We used the public code of the method proposed in [23] for our task. In this model, we use the same multiple window sizes (1,2) to model the tweet.
- **ASC-HDP**: We implemented the method proposed in [54], and we use the same parameters for this task. In this model, it incorporates the user, structure and author information.
- **U-CNN**: U-CNN is a model proposed in this paper which is incorporate the user embedding into the convolutional neural network.
- **UA-CNN**: Different from the model U-CNN, in addition to considering the user embedding, UA-CNN also incorporate the author embedding into the model.
- **SUA-ACNN**: SUA-ACNN is the model proposed in this work, we compute the similarity score as one feature with similarity matrix, and embedded the user's attention interests, finally we concatenate the user embedding, user interests embedding, author embedding, tweet embedding and the similarity score for the retweet prediction.

4.3 Experimental Results

Table 2 shows the comparisons of the proposed method "SUA-ACNN" with the state-of-the-art methods on the constructed evaluation dataset. The "Random" method was the baseline method without taking into account any information. And we can observe that "Random" method achieved the worst performance which is consistent with our assumption. In the method "Ave-SVM" and "Sum-SVM", we use down-sampling method for the problem of positive and negative samples are not balanced, when we don't use the down-sampling method, no samples are predicted to positive in the test data for both the methods 'Ave-SVM" and "Sum-SVM". Comparing the results of "Ave-SVM" and "Sum-SVM" after down-sampling, we can see that "Sum-SVM" could obtain a better performance than "Ave-SVM". This is because "Sum-SVM" could obtain the length information of the tweet. We can observe that the performance of "CNN" is very poor, this is because that applying CNN to model the text directly while ignore the user information is not reasonable. To use CNN for our task, we proposed the "U-CNN" to incorporate the user information through connect the user embedding with the tweet embedding. From the performance of "U-CNN", we can see that "U-CNN" will achieve a better performance than "CNN", which shows the effectively of our method with user information. And comparing the results of "U-CNN" and "Sum-SVM", we can see that the "U-CNN" method can get a better performance. We believe that this is because "U-CNN" with multiple window sizes (1,2) could capture the unigram and bigram information together, and it is also proof the powerful of the CNN for this task. The method "U-CNN" have obtained a better performance in F_1-Score, while in this method, we have not consider the author information. To incorporate the author information, we proposed the method "UA-CNN". From the results of "UA-CNN" and "U-CNN", we can observe that the author information can improve the performance significantly. In the method 'SUA-ACNN", we combine all the information proposed in this paper, from the results of 'SUA-ACNN" and "UA-CNN", we can observe that the similarity score and attention mechanism proposed in this work can achieve a significant improvement. Comparing the F1-score of "SUA-ACNN" with "ASC-HDP", we can see that the proposed model produces a 9.9% relative improvement.

Table 3: Performance on variants of the method.

Methods	**P**	**R**	**F_1**
SUA-ACNN/A	0.730	0.688	0.708
SUA-ACNN/S	**0.775**	0.628	0.694
SUA-ACNN-Random	0.641	**0.766**	0.698
SUA-ACNN	0.733	0.708	**0.721**

Table 3 lists the results of the variant models. "SUA-ACNN" is the proposed model with all the information considered. "SUA-ACNN/A" is the model without the attention mechanism. "SUA-ACNN/S" is the model without considering the similarity score. In the proposed model "SUA-ACNN", we use a cluster method to extract 5 tweets as the interests of the user and use a attention mechanism to select the attention interests, while in the method "SUA-ACNN-Random" we random select 5 tweets as the interests of the user and also use a attention mechanism for the attention interests. From the results of "SUA-ACNN" and "SUA-ACNN/A", we can observe that the performance of "SUA-ACNN" is better than that of "SUA-ACNN/A", The results demonstrate that the attention mechanism can benefit the performance. From the results of "SUA-ACNN" and "SUA-ACNN/S", we can observe that the model with the similarity score can achieve a significant better performance than the model without the similarity score, which proves the effective of the similarity score. Through comparing

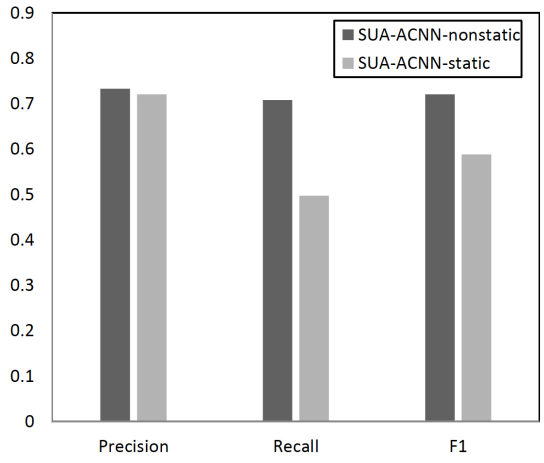


Figure 4: Performance on static vs nonstatic with word2vec.

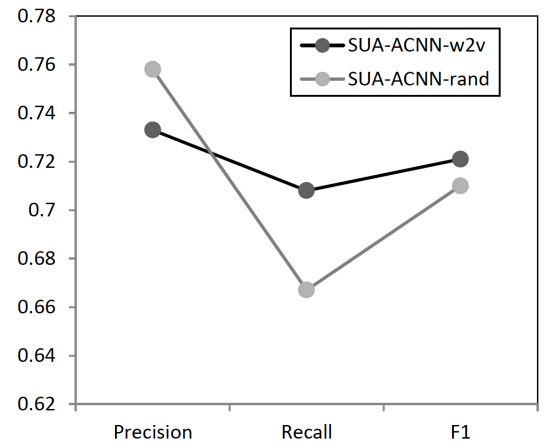


Figure 5: Performance on word2vec vs rand.

the results of "SUA-ACNN" and "SUA-ACNN-Random", we can observe that the cluster method used to extract the user interests can improve the performance in the F1-score significantly. The results demonstrate that all the factors we proposed in the previous section can improve the prediction performance.

In Figure 4, "SUA-ACNN-nonstatic" represents the vectors of the words and users that will be fine-tuned during the training process. "SUA-ACNN-static" represents the vectors will not be changed in the training process. Through comparing the results of "SUA-ACNN-nonstatic" with "SUA-ACNN-static", we can see that "SUA-ACNN-nonstatic" model could achieve a better F1-score performance than "SUA-ACNN-static" model. This result demonstrates that fine-tuning the vectors can improve the performance. The reason is that a "SUA-ACNN-nonstatic" model can make the vectors more specific for the task.

In Figure 5, "SUA-ACNN-rand" represents the word vectors that are randomly initialized. "SUA-ACNN-w2v" represents the vectors that are pre-trained from Google News. From the results of random vectors and word vectors, we can observe that the pre-trained vectors are effective for our task. This is because the pre-training can make the vectors representation more meaningful.

We have the intuition that if the user retweet more times, we can accurately estimate their behaviors. To investigate this intuition, we split the users into three groups based on the number of tweets retweeted by them. Figure 6 shows the results. We can see that the more retweet times of the users, the more accuracy we can get.

Table 4: Performance on variants of the window size.

Window Size	P	R	F_1
1	0.747	0.685	0.715
2	0.755	0.679	0.715
3	0.746	0.694	0.719
4	0.803	0.592	0.682
1-2	0.733	0.708	0.721
1-2-3	0.781	0.634	0.700
1-2-3-4	0.796	0.592	0.679

4.4 Parameters Analysis

Table 4 lists the results of using different window sizes for the filters in the tweet encoding process. We first set the window sizes to 1, 2, 3 and 4 to observe the influence of different window sizes. Then we set multiple window sizes to (1,2), (1,2,3) and (1,2,3,4) to investigate the effectiveness of window size combinations. By using different multiple window sizes, we obtained the best performance when the window sizes were (1,2). From the results of window sizes equals to 1, 2, 3 and 4, we can observe that the best performance will obtained when the window size equals to 3. Comparing the results of window size equals to 3 with window size equals to 1 or 2. The performance is better when window size equals to 3, Because the different window sizes 1, 2 and 3 correspond to the encoding for the unigrams, bigrams and trigrams of the tweets respectively, the trigrams could capture more context information. Comparing the results of window size equals to 3 and 4, we can see that the performance decrease with the window size increase, we believe that a complete semantic unit generally comprises less than three words, when the window size too large will introduce the influence of noise. Comparing the results of multiple window sizes of (1,2) with those of the single window sizes 1, 2, 3 and 4, we can observe that we obtain the best performance with the multiple window sizes (1,2). This is because the multiple window sizes (1,2) can combine the unigrams and bigrams. While comparing the results of multiple window sizes of (1,2),(1,2,3) and (1,2,3,4), we can see that the combination of unigram and bigram can achieve the best performance, we believe that unigram and bigram features complement each other, while more grams feature introduce more noise than supplement.

Figure 7 lists the influence of the batch size. We varied the batch size from 10 to 100. From the figure, we can observe that the performance of the proposed model improves with an increase in the batch size from 10 to 40. However, the performance is decreasing when the batch size is greater than 40. We obtain the best performance when the batch size equals to 40. This is because batch represents the sampling realization of the full samples. Full samples may obtain the local optimal solution, while different batch size will introduce different level gradient correction for the noise, and more likely to search for the optimal value. In this paper, we set the batch size to 40.

Table 5 lists the results of training with different number of iteration. We varied the number of iteration from 1 to 10. We preserve the model which achieve the best performance on the validation data for each iteration. From the results of iteration equals to 4 and 5, we can observe that the performances are

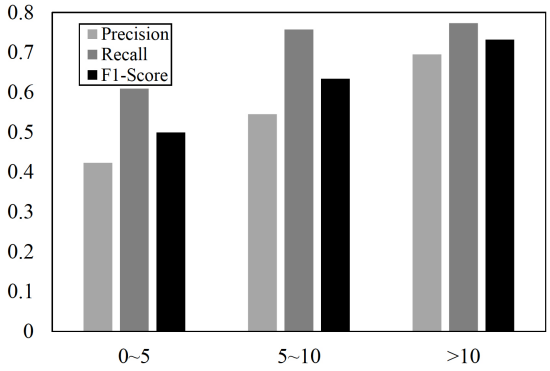


Figure 6: The influence of the number of retweet of the users.

Table 5: Performance on variants of the iteration.

Iteration	P	R	F_1
1	0.805	0.467	0.591
2	0.816	0.502	0.622
3	0.753	0.645	0.695
4	0.778	0.630	0.696
5	0.778	0.630	0.696
6	0.752	0.673	0.711
7	0.733	0.708	0.721
8	0.733	0.708	0.721
9	0.733	0.708	0.721
10	0.733	0.708	0.721

the same. This is because when the iteration equals to 5, the performance on the validation data poorer than iteration equals to 4, we will not update the model preserved. From the results of iteration times more than 7. We can observe that the performance is convergence when the iteration less than 10, it is shows that we don't need many iteration times for the model convergence.

5. CONCLUSION

In this paper, we proposed a novel attention-based deep neural network to obtain the user's attention interests from an attention-based neural network. In this method, we compute the user's attention interests through two steps: from the first step, we cluster the history tweets of the user into some clusters and we use the central tweets of the clusters to represent all the different interests of the user. From the second step, we compute the attention weight for the each interests by an attention layer. And in our model, we compute the similarity score between the user's attention interests and the tweet through a similarity matrix. Then we combine the user embedding, the user's attention interests embedding, the similarity score, the tweet embedding and the author embedding into a fixed feature vector to predict retweet behavior.

To evaluate the proposed method, we collected a large number of tweets and their corresponding social networks from Twitter. Experimental results demonstrate that (1) the proposed method can achieve better performance than state-of-the-art methods. The relative improvement of the proposed SUA-ACNN over ASC-HDP is about 9.9% in terms of the F1-Score. (2) The user embedding, the author embedding, the similarity score and the user's attention interests can each significantly improve the performance, and we obtain the best performance when these are integrated in our model.

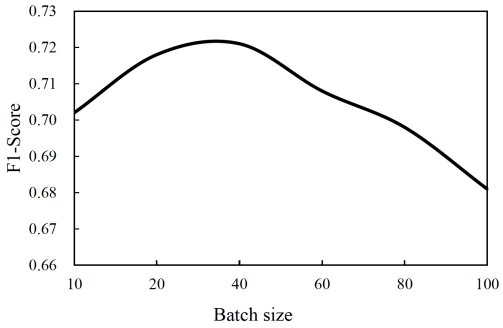


Figure 7: Performance on variants of batch size.

6. ACKNOWLEDGEMENT

The authors wish to thank the anonymous reviewers for their helpful comments. This work was partially funded by National Natural Science Foundation of China (No. 61532011, 61473092, and 61472088), the National High Technology Research and Development Program of China (No. 2015AA015408).

7. REFERENCES

[1] O. Abdel-Hamid, L. Deng, and D. Yu. Exploring convolutional neural network structures and optimization techniques for speech recognition. In *INTERSPEECH*, pages 3366–3370, 2013.

[2] P. Achananuparp, E.-P. Lim, J. Jiang, and T.-A. Hoang. Who is retweeting the tweeters? modeling, originating, and promoting behaviors in the twitter network. *TMIS*, 3(3):13, 2012.

[3] D. Bahdanau, J. Chorowski, D. Serdyuk, P. Brakel, and Y. Bengio. End-to-end attention-based large vocabulary speech recognition. *arXiv preprint arXiv:1508.04395*, 2015.

[4] B. Bi and J. Cho. Modeling a retweet network via an adaptive bayesian approach. In *Proceedings of the 25th International Conference on World Wide Web*, pages 459–469. International World Wide Web Conferences Steering Committee, 2016.

[5] P. Blunsom, E. Grefenstette, N. Kalchbrenner, et al. A convolutional neural network for modelling sentences. In *Proceedings of ACL*, 2014.

[6] A. Bordes, J. Weston, and N. Usunier. Open question answering with weakly supervised embedding models. In *Machine Learning and Knowledge Discovery in Databases*, pages 165–180. Springer, 2014.

[7] D. Boyd, S. Golder, and G. Lotan. Tweet, tweet, retweet: Conversational aspects of retweeting on twitter. In *System Sciences (HICSS), 2010 43rd Hawaii International Conference on*, pages 1–10. IEEE, 2010.

[8] E. F. Can, H. Oktay, and R. Manmatha. Predicting retweet count using visual cues. In *Proceedings of the 22nd ACM international conference on information & knowledge management*, pages 1481–1484. ACM, 2013.

[9] K. Chen, J. Wang, L.-C. Chen, H. Gao, W. Xu, and R. Nevatia. Abc-cnn: An attention based convolutional neural network for visual question answering. *arXiv preprint arXiv:1511.05960*, 2015.

[10] J. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio. Attention-based models for speech recognition. *arXiv preprint arXiv:1506.07503*, 2015.

[11] P. Cui, S. Jin, L. Yu, F. Wang, W. Zhu, and S. Yang. Cascading outbreak prediction in networks: a data-driven approach. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 901–909, 2013.

[12] P. Cui, F. Wang, S. Liu, M. Ou, S. Yang, and L. Sun. Who should share what?: item-level social influence prediction for users and posts ranking. In *Proceeding of the International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2011, Beijing, China, July*, pages 185–194, 2011.

[13] M. Denil, A. Demiraj, N. Kalchbrenner, P. Blunsom, and N. de Freitas. Modelling, visualising and summarising documents with a single convolutional neural network. *arXiv preprint arXiv:1406.3830*, 2014.

[14] A. Echihabi and D. Marcu. A noisy-channel approach to question answering. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*, pages 16–23. Association for Computational Linguistics, 2003.

[15] W. Feng and J. Wang. Retweet or not?: personalized tweet re-ranking. In *Proceedings of the sixth ACM international conference on Web search and data mining*, pages 577–586. ACM, 2013.

[16] J. Gao, L. Deng, M. Gamon, X. He, and P. Pantel. Modeling interestingness with deep neural networks, Dec. 17 2015. US Patent 20,150,363,688.

[17] K. Gregor, I. Danihelka, A. Graves, and D. Wierstra. Draw: A recurrent neural network for image generation. *arXiv preprint arXiv:1502.04623*, 2015.

[18] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012.

[19] L. Hong, O. Dan, and B. D. Davison. Predicting popular messages in twitter. In *Proceedings of the 20th international conference companion on World wide web*, pages 57–58. ACM, 2011.

[20] B. Hu, Z. Lu, H. Li, and Q. Chen. Convolutional neural network architectures for matching natural language sentences. In *Advances in Neural Information Processing Systems*, pages 2042–2050, 2014.

[21] S. Ji, W. Xu, M. Yang, and K. Yu. 3d convolutional neural networks for human action recognition. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 35(1):221–231, 2013.

[22] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei. Large-scale video classification with convolutional neural networks. In *CVPR*, pages 1725–1732. IEEE, 2014.

[23] Y. Kim. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*, 2014.

[24] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[25] A. Kupavskii, L. Ostroumova, A. Umnov, S. Usachev, P. Serdyukov, G. Gusev, and A. Kustarev. Prediction of retweet cascade size over time. In *Proceedings of the 21st ACM international conference on Information and knowledge management*, pages 2335–2338. ACM, 2012.

[26] Z. Luo, M. Osborne, S. Petrovic, and T. Wang. Improving twitter retrieval by exploiting structural information. In *AAAI*, 2012.

[27] Z. Luo, M. Osborne, J. Tang, and T. Wang. Who will retweet me?: finding retweeters in twitter. In *Proceedings of SIGIR*, pages 869–872. ACM, 2013.

[28] M.-T. Luong, H. Pham, and C. D. Manning. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*, 2015.

[29] T. Mikolov, M. Karafiát, L. Burget, J. Cernockỳ, and S. Khudanpur. Recurrent neural network based language model. In *INTERSPEECH*, 2010.

[30] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[31] V. Mnih, N. Heess, A. Graves, et al. Recurrent models of visual attention. In *NIPS*, 2014.

[32] M. Oquab, L. Bottou, I. Laptev, and J. Sivic. Learning and transferring mid-level image representations using convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1717–1724, 2014.

[33] H.-K. Peng, J. Zhu, D. Piao, R. Yan, and Y. Zhang. Retweet modeling using conditional random fields. In *Data Mining Workshops (ICDMW), 2011 IEEE 11th International Conference on*, pages 336–343. IEEE, 2011.

[34] S. Petrovic, M. Osborne, and V. Lavrenko. Rt to win! predicting message propagation in twitter. In *ICWSM*, 2011.

[35] R. Pfitzner, A. Garas, and F. Schweitzer. Emotional divergence influences information spreading in twitter. *ICWSM*, 12:2–5, 2012.

[36] P. H. Pinheiro and R. Collobert. Recurrent convolutional neural networks for scene parsing. *arXiv preprint arXiv:1306.2795*, 2013.

[37] A. Severyn and A. Moschitti. Learning to rank short text pairs with convolutional deep neural networks. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 373–382. ACM, 2015.

[38] S. Sharma, R. Kiros, and R. Salakhutdinov. Action recognition using visual attention. *arXiv preprint arXiv:1511.04119*, 2015.

[39] Y. Shen, X. He, J. Gao, L. Deng, and G. Mesnil. Learning semantic representations using convolutional neural networks for web search. In *Proceedings of the companion publication of the 23rd international conference on World wide web companion*, pages 373–374. International World Wide Web Conferences Steering Committee, 2014.

[40] R. Socher, C. C. Lin, C. Manning, and A. Y. Ng. Parsing natural scenes and natural language with recursive neural networks. In *Proceedings of ICML*, pages 129–136, 2011.

[41] E. Spiro, C. Irvine, C. DuBois, and C. Butts. Waiting for a retweet: modeling waiting times in information propagation. In *2012 NIPS workshop of social networks and social media conference. http://snap. stanford. edu/social2012/papers/spiro-dubois-butts. pdf. Accessed*, volume 12, 2012.

[42] S. Stieglitz and L. Dang-Xuan. Political communication and influence through microblogging–an empirical analysis of sentiment in twitter messages and retweet behavior. In *2012 45th Hawaii International Conference on System Sciences*.

[43] B. Suh, L. Hong, P. Pirolli, and E. H. Chi. Want to be retweeted? large scale analytics on factors impacting retweet in twitter network. In *Social computing (socialcom), 2010 ieee second international conference on*, pages 177–184. IEEE, 2010.

[44] T. Wang, D. J. Wu, A. Coates, and A. Y. Ng. End-to-end text recognition with convolutional neural networks. In *Pattern Recognition (ICPR), 2012 21st International Conference on*, pages 3304–3308. IEEE, 2012.

[45] T. Xiao, Y. Xu, K. Yang, J. Zhang, Y. Peng, and Z. Zhang. The application of two-level attention models in deep convolutional neural network for fine-grained image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 842–850, 2015.

[46] F. Xiong, Y. Liu, Z.-j. Zhang, J. Zhu, and Y. Zhang. An information diffusion model based on retweeting mechanism for online social media. *Physics Letters A*, 376(30):2103–2108, 2012.

[47] K. Xu, J. Ba, R. Kiros, A. Courville, R. Salakhutdinov, R. Zemel, and Y. Bengio. Show, attend and tell: Neural image caption generation with visual attention. *arXiv preprint arXiv:1502.03044*, 2015.

[48] M.-C. Yang, J.-T. Lee, S.-W. Lee, and H.-C. Rim. Finding interesting posts in twitter based on retweet graph analysis. In *Proceedings of SIGIR*, 2012.

[49] Z. Yang, J. Guo, K. Cai, J. Tang, J. Li, L. Zhang, and Z. Su. Understanding retweeting behaviors in social networks. In *Proceedings of CIKM*. ACM, 2010.

[50] T. R. Zaman, R. Herbrich, J. Van Gael, and D. Stern. Predicting information spreading in twitter. In *Workshop on computational social science and the wisdom of crowds, NIPS*, 2010.

[51] M. D. Zeiler. Adadelta: An adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.

[52] J. Zhang, B. Liu, J. Tang, T. Chen, and J. Li. Social influence locality for modeling retweeting behaviors. In *Proceedings of AAAI*, 2013.

[53] J. Zhang, J. Tang, J. Li, Y. Liu, and C. Xing. Who influenced you? predicting retweet via social influence locality. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 9(3):25, 2015.

[54] Q. Zhang, Y. Gong, Y. Guo, and X. Huang. Retweet behavior prediction using hierarchical dirichlet process. In *AAAI*, 2015.