Modeling the Long-term Post History for Personalized Hashtag Recommendation

Minlong Peng, Yaosong Lin, Lanjun Zeng, Tao Gui, and Qi Zhang*

School of Computer Science, Fudan University, Shanghai, China
{mlpeng16, yslin18, ljzeng18, tgui16, qz}@fudan.edu.cn

Abstract. Hashtag recommendation aims to recommend hashtags when social media users show the intention to insert a hashtag by typing in the hashtag symbol "#" while writing a microblog. Previous methods usually considered the textual information of the post itself or only fixed-length short-term post history. In this paper, we propose to model the long-term post histories of user with a novel neural memory network called the Adaptive neural MEmory Network (AMEN). Compared with existing memory networks, AMEN was specially designed to combine both content and hashtag information from historical posts. In addition, AMEN contains a mechanism to deal with out-of-memory situations. Experimental results on a dataset of Twitter demonstrated that the proposed method significantly outperforms the state-of-the-art methods.

Keywords: Hashtag recommendation · Long-term post history · Neural memory network

1 Introduction

Along with the rapid development of social media, microblogging has experienced tremendous success as a news medium. Every day, hundreds of millions of posts are created. To facilitate the navigation of this huge knowledge base, microblogging services encourage users to insert hashtags, which start with the "#" symbol (e.g., "#CCL2019", "#MentalHealth"), into posts to concisely indicate an object or categorize their posts. However, due to the increasing number of hashtags, it is not easy for authors to find appropriate hashtags matching their intention. Therefore, recommending hashtags to users is imperative.

In recent years, a variety of methods have been proposed to perform hashtag recommendation These methods can be generally organized into two groups. The first group of methods treat posts of different users without distinction. They mainly focus on modeling the textual content of posts, which is traditionally dominated by topic-model-based methods [5, 6, 18, 7]. This dominance has been overturned by a resurgence of interest in deep neural network based approaches [3, 8, 15]. The second group of methods additionally model the user's personal information, which is commonly extracted from their post history [21–23, 11, 13, 20]. Zhang et al. [22] proposed the TPLDA model for this purpose. Their model grouped posts by user and introduced an additional parameter for each

user into the LDA model. Huang et al. [11] proposed a hierarchical end-to-end memory model (HMemN2N) to encode the post history. For each user, their model constructed the memory using the latest five historical posts and then recursively accessed it with the current post content for the recommendation. These existing methods have already demonstrated the informativeness of the post history. However, all of the above methods can only model short and fixed-length post histories.

In this work, we investigate the effectiveness of utilizing the long-term post history for hashtag recommendation. Modeling long-term post history is structurally inapplicable for the previous methods because of the incrementally increasing size of records. Inspired by the success of neural memory networks in modeling long-term dependency of sequences [1, 9, 19], we propose a novel neural memory network called the Adaptive neural MEmory Network (AMEN), to tackle the challenge. AMEN was specially designed to treat the content and the hashtag as two different views of posts, rather than treat hashtags as common words. The two different representations are simultaneously combined and encoded into memory. We argue that this design has several benefits. First, it does not require that the representations of words and hashtags in the same vector space. Second, it can reduce the size of the word vocabulary used to represent hashtags. Finally, it highlights the hashtag information, making its modeling more flexible. Despite the significance of post history, considering the immediacy of microblogging, a user may post a tweet about a brand new topic that is almost independent of historical posts. That is what we call the out-of-memory (OOM) problem. To address this situation, we designed a novel gate mechanism. The main contributions of this work are summarized as follows:

- We proposed a novel neural memory network that combines both textual content and hashtags to model users' long-term post history for personalized hashtag recommendation tasks.
- We designed a gate mechanism to deal with OOM situations where the hashtag usage is almost independent of previous posts.
- Comprehensive experiments and quantitative studies demonstrate the effectiveness of the proposed model on a dataset collected from Twitter.

2 Proposed Model

In this work, we formulate the task of recommending hashtags as a matching-based problem. Given a user $\mathbf{u}$ with post history sequence $[(p_1, H_1), (p_2, H_2), \cdots, (p_{t-1}, H_{t-1})]$, for the t^{th} post, our task is to rank the candidate hashtags $H = \{H^1, H^2, \cdots, H^n\}$. To address this problem, we proposed the Adaptive neural MEmory Network (AMEN). The general architecture of the proposed model is depicted in Figure 1, which is specially designed to highlight the core of this model in modeling the long-term post history.

The proposed model could be separated into three parts. First, we employ a convolution neural network and a recurrent neural network to obtain representation of post content and hashtag. Second, we model users' long-term post history

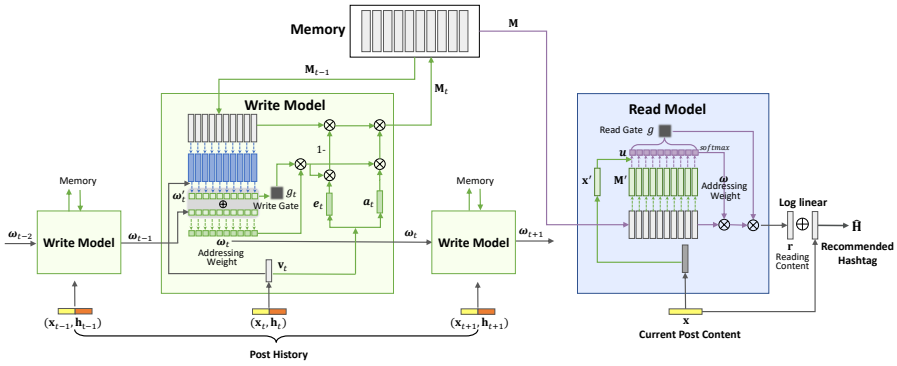

Fig. 1. General architecture of the proposed model.

with the core memory network. Lastly, we combine the current post content with memory augmented information to rank candidate hashtags.

2.1 Representation Learning

Content Representation We utilize a *one-layer convolution neural network* to obtain the post content representation. It is a variant of the traditional convolution network proposed by Kim et al.[12]. Specifically, let $\mathbf{w}_i \in R^{k_w}$ be the k_w-dimensional word vector, corresponding to the i^{th} word of the post. Therefore, A post of length n is represented as:

$$\mathbf{p} = \left[\mathbf{w}_1, \mathbf{w}_2, \cdots, \mathbf{w}_n\right] \tag{1}$$

The one-layer CNN takes the dot product of the filter $\mathbf{m} \in R^{k_w \times h}$ with each h-gram in $\mathbf{p}$ to obtain a sequence $\mathbf{s}$.

$$\mathbf{s}_i = f(\mathbf{m} \cdot \mathbf{p}_{i:i+h-1} + b). \tag{2}$$

Here, $b \in R$ is a bias term, and f is the hyperbolic tangent function. This filter is applied to each possible window of words in the sequence $\{\mathbf{p}_{1:h}, \mathbf{p}_{2:h+1}, \cdots, \mathbf{p}_{n-h+1:n}\}$ to produce a feature map:

$$\mathbf{s} = [\mathbf{s}_1, \mathbf{s}_2, \cdots, \mathbf{s}_{n-h+1}] \tag{3}$$

To deal with various post lengths, max pooling is applied over the feature map. By extending this operation to multiple filters with various window sizes, we obtain multiple features:

$$\mathbf{x} = \left[\max(\mathbf{s}^1) \cdots \max(\mathbf{s}^d)\right] \tag{4}$$

Here d is the filter number and $\mathbf{s}^i$ is the feature map extracted with the i^{th} filter. We use these features as the post content representation, denoted by $\mathbf{x}$.

Hashtag Representation Most of the previous hashtag recommendation systems treat the hashtags as independent meaningless categories and represent them with orthogonal one-hot vectors or randomly initialized dense vectors. We argue that there are two drawbacks of this practice. First, it neglects the semantic relationship between hashtags. For examples, "Christmas" and "ChristmasEve" are semantically close, so the representation of them should naturally be close in the vector space. Second, in this practice, the size of the hashtag set is unfixed. Different from previous work, we apply a recurrent neural network to the character sequence of hashtags to capture the underlying semantic and obtain their vector representations. The formal definition is specified as follows:

$$\mathbf{h}_t = \tanh(\mathbf{W}_h \cdot \mathbf{h}_{t-1} + \mathbf{W}_c \cdot \mathbf{c}_t + \mathbf{b}_c) \tag{5}$$

where $\mathbf{c}_t \in R^{d_c}$ is the vector representation of the t^{th} character of the hashtag $\mathbf{H}$, $\mathbf{W}_{\mathrm{h}} \in \mathbb{R}^{d_h \times d_h}$, $\mathbf{W}_{\mathrm{c}} \in \mathbb{R}^{d_h \times d_c}$ and $\mathbf{b}_{\mathrm{c}} \in \mathbb{R}^{d_h}$ are trainable parameters. We take the final hidden state $\mathbf{h_n} \in \mathbf{R}^{d_h}$ to represent $\mathbf{H}$. In the following sections, we refer $\mathbf{h}$ to this vector representation if without further explanation.

2.2 Adaptive neural MEmory Network

In this section, we describe the core parts of the proposed model AMEN. Intuitively, the proposed memory network consists of a memory module, a write module and a read module, which will be introduced in detail.

Memory Module The memory module consists of several dense vector cells for storing historical post information. Technically, let $\mathbf{M} \in \mathbb{R}^{N_m \times d}$ be the contents of memory matrix for each user, where N_m denotes the number of memory locations, and d is the vector dimension at each location, while $\mathbf{M}_t$ represents the memory state after the write process at time step t.

Note that the memory is empty at the beginning. If no further processing is applied, the memory cells will be read and written equally, acting just like a single cell. To avoid this, in every epoch, we write the first N_m samples of each user into the memory, rather than initialize all memory cells with zero vectors.

Write Module The write module is designed to encode the post history one-by-one into the memory module. During the writing process, both the post content representation $\mathbf{x}_t$ and the hashtag representation $\mathbf{h}_t$ are written into the memory at time step t. More specifically, we construct the post content $\mathbf{v}_t$ with the combination of $\mathbf{x}_t$ and $\mathbf{h}_t$ as follows:

$$\mathbf{v}_t = \mathbf{MLP}(\mathbf{x}_t \oplus \mathbf{h}_t;\ \boldsymbol{\theta}_o). \tag{6}$$

This MLP consists of two non-linear layers with sigmoid and hyperbolic tangent activation, respectively, and $\boldsymbol{\theta}_o$ denotes trainable parameters. Then, we generate

the unnormalized addressing weight $\mathbf{u}'_t(i)$ by similarity measure between the $\mathbf{v}_t$ with each memory vector $\mathbf{M}_{t-1}(i)$, i denoting the index of memory locations:

$$\mathbf{u}'_t(i) = K[\mathbf{v}_t, \mathbf{M}_{t-1}(i)]. \tag{7}$$

where

$$K[\mathbf{x}, \mathbf{y}] = \frac{\mathbf{x}^T \mathbf{y}}{||\mathbf{x}||^2}.$$

Note that this differs from the conventional NTM model[9] in that we omit the norm of the memory matrix in the denominator, so as to preserve the quantity information of each memory cell. After that, we apply a softmax non-linear operation to obtain the normalized addressing weight $\boldsymbol{\omega}'_t$:

$$\boldsymbol{\omega}'_t(i) = \frac{\exp(\mathbf{u}'_t(i))}{\sum_j^{\mathrm{N_m}} \exp(\mathbf{u}'_t(j))}. \tag{8}$$

where N_m is the size of the memory module. By combining $\boldsymbol{\omega}'_t$ with the writing weights of the previous time $\boldsymbol{\omega}_{t-1}$ using a recurrent framework, we obtain the addressing weight $\boldsymbol{\omega}_t$ as follows:

$$\begin{aligned} \mathbf{u}_t(i) &= \mathbf{W}_\omega \left[\boldsymbol{\omega}_{t-1} \oplus \boldsymbol{\omega}'_t\right] + b_\omega \\ \boldsymbol{\omega}_t(i) &= \frac{\exp(\mathbf{u}_t(i))}{\sum_j^{\mathrm{N_m}} \exp(\mathbf{u}_t(j))}. \end{aligned} \tag{9}$$

In addition, considering that a user may re-tweet some posts for particular purposes which are not highly related to their interests, we further generate an writing gate g_t to determine whether or not the current example should be written into the memory, based on the unnormalized address weight $\mathbf{u}'_t$:

$$\begin{aligned} \mathbf{z}_t &= \mathrm{sort}(\mathbf{u}'_t) \\ g_t &= \sigma(\mathbf{s}_\alpha[\mathbf{z}_t \oplus (\mathbf{z}_t - \bar{\mathbf{z}}_t)^2]), \end{aligned} \tag{10}$$

where the sort function takes an array as input and returns its ordered counterpart, $\oplus$ represents the concatenating operation, $\bar{\mathbf{z}}_t$ is the mean value of $\mathbf{z}_t$, and $\mathbf{s}_\alpha \in R^{2N}$ is a trainable user-specific gate vector. Here, the sorting operation is critical because of the randomness inherent in the memory reading and writing operations.

Next, we obtain the content to erase from memory $\mathbf{e}_t$ and the content to write into memory $\mathbf{a}_t$, from the combination of $\mathbf{x}_t$ and $\mathbf{h}_t$:

$$\begin{aligned} \mathbf{e}_t &= \sigma(\mathbf{W}_\mathbf{e}(\mathbf{x}_t \oplus \mathbf{h}_t) + b_e), \\ \mathbf{a}_t &= \tanh(\mathbf{W}_\mathbf{a}(\mathbf{x}_t \oplus \mathbf{h}_t) + b_a). \end{aligned} \tag{11}$$

Finally, we update the memory cells formulated as follows:

$$\mathbf{M}_t(i) = \mathbf{M}_{t-1}(i) \left[\mathbf{1} - g_t \boldsymbol{\omega}_t(i) \mathbf{e}_t\right] + g_t \boldsymbol{\omega}_t(i) \mathbf{a}_t. \tag{12}$$

Read Module The read model is designed to access the memory with the current post content and output a vector representation of the memory, which is further used to recommend hashtags. However, during the reading process, only the post content representation $\mathbf{x}$ of the post $\mathbf{p}$ can be used for addressing. As we mentioned above, the memory contains both the information of content and hashtag of previous posts. Consequently, the vector space of the memory content $\mathbf{M}(i)$ is not consistent with the space of $\mathbf{x}$. In order to utilize the memory, we first incorporate a transformation process to map the input and the memory cells into a consistent space, formulated as follows:

$$\begin{aligned} \mathbf{x}' &= \mathbf{MLP}(\mathbf{x};\ \boldsymbol{\theta}_p), \\ \mathbf{M}'(i) &= \mathbf{MLP}(\mathbf{M}(i);\ \boldsymbol{\theta}_q), \end{aligned} \tag{13}$$

where **MLP** is a two-layer perceptron with *relu* and *tanh* activation functions. We then obtain an unnormalized addressing weight $\mathbf{u}'$:

$$\mathbf{u}(i) = K[\mathbf{x}', \mathbf{M}'(i)], \tag{14}$$

Based on that, we generate the normalized addressing weight $\boldsymbol{\omega}(i)$:

$$\boldsymbol{\omega}(i) = \frac{\exp(\mathbf{u}(i))}{\sum_j^{\mathrm{N_m}} \exp(\mathbf{u}(j))}, \tag{15}$$

The addressing weight $\boldsymbol{\omega}(i)$ can now be used to produce an output of the reading module. However, instead of directly using the addressing weight $\boldsymbol{\omega}(i)$ to obtain memory output $\mathbf{r}$, we consider the out-of-memory situations to enhance our read module in the following.

Out-of-memory Mechanism Despite the significance of post history, there is a risk of the out-of-memory situations of hashtag recommendation task, where users may write a post about events that just happened. In this case, we hope to ignore the memory and perform the recommendation based merely on the post content itself. With this consideration, we design a mechanism to automatically detect the out-of-memory situations based on the global view of the input and the memory. Mathematically, we generate a gate g to indicate whether it should depend on the memory or not, as follows:

$$\begin{aligned} \mathbf{z} &= \mathrm{sort}(\mathbf{u}(i)) \\ g &= \sigma(\mathbf{s}_\beta[\mathbf{z} \oplus (\mathbf{z} - \bar{\mathbf{z}})^2]), \end{aligned} \tag{16}$$

where $\bar{z}$ is the mean value of $\mathbf{z}$, and $\mathbf{s}_\beta \in R^{2N}$ is a trainable user-specific gate vector. Finally, based on the designed gate g and the addressing weight $\boldsymbol{\omega}(i)$, the memory output $\mathbf{r}$ is given by:

$$\mathbf{r} = g \cdot \sum_i^{N_m} \mathbf{M}(i)\boldsymbol{\omega}(i). \tag{17}$$

2.3 Recommendation

Given a post $\mathbf{p}$, the hashtag recommendation of the proposed model is performed based on both the post content $\mathbf{x}$ and the memory augmented output $\mathbf{r}$ using a log-linear model. We redefine the score function of hashtag $\mathbf{h}_i$:

$$\text{score}(\mathbf{p}, \mathbf{h}_i|\mathbf{M}) = \exp(\mathbf{x}^T\mathbf{h}_i + g \cdot \mathbf{r}^T\mathbf{h}_i), \tag{18}$$

where $\mathbf{M}$ denotes the memory content corresponding to $\mathbf{x}$. We apply a softmax operation to obtain the probability of recommending $\mathbf{h}_i$, namely:

$$\mathbf{p}(\mathbf{h}_i|\mathbf{p}, \mathbf{M}) = \frac{\exp\left(\text{score}(\mathbf{p}, \mathbf{h}_i)\right)}{\sum_{j=0}^{\mathrm{N}_h} \exp\left(\text{score}(\mathbf{p}, \mathbf{h}_j)\right)}, \tag{19}$$

where N_h is the size of candidate hashtag set. Top-k hashtags with the highest probability are recommended for post $\mathbf{p}$.

2.4 Training and Inference

The training objective function of the model is defined as:

$$J = \frac{1}{\sum_u (\mathrm{N}_u - \mathrm{N}_m)} \sum_u \sum_{t=\mathrm{N}_m}^{\mathrm{N}_u} -\log p(\mathbf{H}_t|\mathbf{p}_t, \mathbf{M}_{t-1}; \mathbf{\Theta}), \tag{20}$$

where N_u is the number of history posts for user $\mathbf{u}$, N_m is the size of memory locations, and $\mathbf{\Theta}$ denotes the parameter set. We sample $\mathbf{N}_m$ posts of each user to initialize the memory, so these posts are excluded from the calculation of training cost.

During inference, for each user, we fix the memory content $\mathbf{M}$ corresponding to the state after writing last training example. For user $\mathbf{u}$ and his/her post $\mathbf{p}$, the probability of hashtag $\mathbf{H}_i$ as the recommendation candidate is defined as:

$$\mathbf{H}_i = p(\mathbf{H}_i|\mathbf{p}, \mathbf{M}_{\mathrm{N}_u}; \mathbf{\Theta}). \tag{21}$$

Here $\mathbf{M}_{\mathrm{N}_u}$ denotes the memory state after writing the last training example.

3 Experiment

3.1 Dataset

Most of previous studies only consider the content of microblogs for the hashtag recommendation, and there is no user and time information within their datasets. To evaluate the proposed model, we constructed a new dataset from Twitter.

We crawled tweets posted within 1/1/2015 to 2/28/2015 and removed users who have less than 30 or more than 300 posts within the duration. The lower bound was used to ensure that our system could learn the user-specific parameters well, and the upper bound was to filter out noisy users like advertisers. From these users, we selected 2,000 users and extracted all the tweets posted by them for training. For development and testing dataset, we crawled tweets posted by these users during 3/1/2015 to 3/10/2015, and applied the same preprocessing. Statistical information about this dataset is listed in Table 1.

Table 1. Statistics of the dataset in experiments.

Item	Train	Develop	Test
#User	2,000	1,000	1,000
#Tweet	127,846	8,086	9,190
#Example	187,247	13,174	13,946
#Hashtag	3,104	1,936	1,952
#Hashtag / User	23.54	6.28	6.29
#Length / Example	13.30	13.26	13.04

3.2 Methods for Comparison

We first compared the proposed model with several baselines and state-of-the-art methods, including methods that ignore the post history and those that only model the short-term post history.

The history-independent models includes the translation model **IBM1**[17], the topical word alignment model **TopicWA**[5], the discriminated lstm model **Tweet2Vec**[3]. And the history-dependent models includes the attention-based LSTM model **LSTM-Attention** [16], the user-specific topic model **TPLDA**[22], the end-to-end memory network **HMemN2N**[11], and the naive neural turing machine model **NTM**[9] which does not separately model the hashtag from the post content and not deal with the out-of-memory problem of this task.

Then to explore the effectiveness of several main components of the proposed model, we implemented the two variants of AMEN. The first variant only writes the post contents into the memory without the hashtag history, called **AMEN w/o Post History**. This was designed to validate the informativeness of historical hashtags. The second one, called **AMEN w/o OOM**, ignores the out-of-memory problem for hashtag recommendation as described above. This is designed to verify the effectiveness of the mechanism for dealing with the out-of-memory (OOM) situations.

3.3 Implementation Details

Parameter For the proposed model and its variants, the embedding dimensions of the word, character, and memory cell were set to 300, 50, and 200 respectively. We initialized the word embeddings with Word2vec. Hidden size of the recurrent neural network for hashtag encoding was set to 100. Memory size N_m was set to 5. For the post content encoding, we used 200 filters for each h-gram size $\in \{1, 2, 3, 4\}$. Dropout was applied to word embeddings with a probability of 0.5. We used the Adadelta optimizer for optimization.

Evaluation Metric There are several evaluation metrics for hashtag recommendation, such as $Hits@N$ [18, 14], Precision, Recall and F1 [18, 11]. Here, we choose the $Hits@N$ because in the setting of our work, there is only one ground truth hashtag for every recommendation. A hit occurs when the ground truth hashtag is include among the recommended n hashtags.

$$Hits@(n) = \frac{\text{Number of Hits}}{\text{Recommendation times}}.$$

Table 2. Comparison of our method with state-of-the-art methods and two variants.

Models	Hits@1	Hits@5
IBM1 [17]	0.2322	0.3043
TopicWA [5]	0.3023	0.3975
Tweet2Vec [3]	0.3116	0.4021
LSTM-Attention [16]	0.3413	0.4430
TPLDA [22]	0.2737	0.5359
HMemN2N [11]	0.3843	0.5460
NTM [9]	0.4021	0.5936
AMEN (proposed)	**0.4732**	**0.6744**
AMEN w/o Hashtag History	0.4251	0.6237
AMEN w/o OOM	0.4372	0.6414

3.4 Results and Discussion

Table 2 shows that the proposed model outperforms all state-of-the-art methods, which indicates the effectiveness of AMEN. From this table, we can draw the following conclusions.

First, modeling users' historical posts provides valuable information for recommendations. According to Table 2, the methods modeling post history (e.g., TPLDA, HMemN2N, NTM) generally outperform content-only-based methods (e.g., IBM1, TopicWA, Tweet2Vec), especially on Hits@5. This validates the informativeness of the post history for hashtag recommendation.

Next, for methods considering historical posts, leveraging the long-term post history could bring significant improvements. Compared to HMemN2N, AMEN w/o Hashtag History which models the long-term post history instead of the short-term history, improves Hits@1 and Hits@5 by approximately 4% and 8%, respectively. This empirically verifies our assumption that the long-term post history should be informative.

Lastly, compared to its variants, the performance of AMEN can be obviously boosted with the two specially designed components. On the one hand, comparison of AMEN and AMEN w/o Hashtag History shows that modeling users's historical hashtags bring additional improvements of 5% on Hits@1 and Hits@5, revealing the high informativeness of historical hashtags. On the other hand, by incorporating an OOM gating mechanism, AMEN beats AMEN w/o OOM about 4% on Hits@1 and 3% on Hits@5, which justifies the necessity to deal with OOM situations. Combining the two designs together, AMEN yields 47.32% on Hits@1 and 67.44% on Hits@5, showing noticeable improvements.

Impact of Different Parts of Post History. How much influence each part of the post history has on the final recommendation performance? To explore this, we performed a study on different parts of the post history. For each user, we removed different portions of training data and re-generated the memory content $\mathbf{M}_{\mathrm{N}_u}$ using the trained parameters and the left training data.

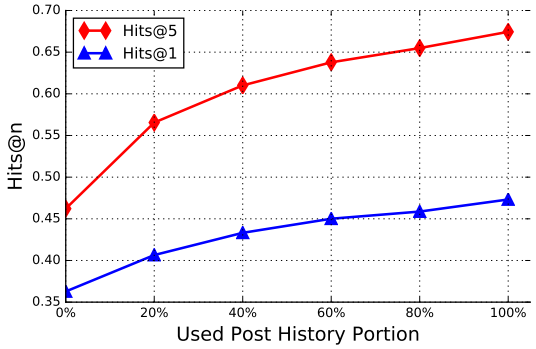

Fig. 2. Performance of the proposed model using different portion of post history.

Figure 2 shows the results of the proposed model using the latest 20%, 40%, 60%, 80%, and 100% of the post history for each user. Here the latest post history refers to historical posts that are posted at the time closest to the testing dataset. From the figure, we can see that the performance of our proposed model continuously increases as the portion of the post history increases. In addition, the improving speed slowly decreases as more historical posts are considered. For example, with the most recent 20% of the post history, the performance increases approximately 4% on Hits@1 and 10% on Hits@5. While additionally using 20%-40% of the post history, the performance only increases almost 3% on Hits@1 and 4% on Hits@5. We argue that the latest 20% of the post history has a greater influence on the testing data than the 20%-40% ones. Similar observations can be obtained from comparisons between other portions.

Impacts of Filter Number. One hyper-parameter we are interested in is the filter number N_f for each n-gram, so we tried different values ranging from 100 to 250. As shown in Table 3, our proposed model performs quite robustly to the variation of N_f over the given range. Moreover, when the value is smaller than 200, increasing this value can sightly improve the general performance. However, when the value reaches 200, increasing this value can even harm the performance because of more serious over-fitting problem.

Table 3. Performance of our proposed model with different filter numbers.

Filter Number	Hits@1	Hits@5
100	0.4454	0.6628
150	0.4663	**0.6768**
200	**0.4732**	0.6744
250	0.4500	0.6637

Impact of Memory Size. We further investigated the impacts of memory size N_m on model performance. Table 4 lists the results for various memory sizes $\mathrm{N}_m \in \{2, 3, 5, 7\}$. We can see that the value has great influence on the perfor-

mance of AMEN. It performed better for $N_m = 3$ and $N_m = 5$. Increasing or decreasing the memory size led to a worse performance. This was not unexpected because if the memory size was too small, it could not effectively remember the post history, while if the memory size was too large, the over-fitting problem would be more serious.

Table 4. Performance of our proposed model with different memory size.

Memory Size	Hits@1	Hits@5
2	0.4483	0.6683
3	0.4698	**0.6757**
5	**0.4732**	0.6744
7	0.4555	0.6674

4 Conclusion

In this work, we addressed the problem of recommending hashtags when a user shows the intention to insert a hashtag by typing the hashtag symbol "#". Compared to existing works that did not consider users' post histories or considered only short- and fixed-length post histories, we proposed to leverage the long-term post history for hashtag recommendation. For this purpose, we proposed a novel memory network to model the long-term post history of users, which was specially designed to combine both content and the hashtag information from historical posts. In addition, we designed a mechanism to address the out-of-memory problem. Experimental results on a dataset collected from Twitter showed that the proposed method achieves state-of-the-art performance.

References

1. Bordes, A., Usunier, N., Chopra, S., Weston, J.: Large-scale simple question answering with memory networks. arXiv preprint arXiv:1506.02075 (2015)
2. Dey, K., Shrivastava, R., Kaushik, S., Subramaniam, L.V.: Emtagger: a word embedding based novel method for hashtag recommendation on twitter. arXiv preprint arXiv:1712.01562 (2017)
3. Dhingra, B., Zhou, Z., Fitzpatrick, D., Muehl, M., Cohen, W.W.: Tweet2vec: Character-based distributed representations for social media. arXiv preprint arXiv:1605.03481 (2016)
4. Ding, Z., Qiu, X., Zhang, Q., Huang, X.: Learning topical translation model for microblog hashtag suggestion. In: Proceedings of the 22nd International Joint Conference on Artificial Intelligence (IJCAI 2013) (2013)
5. Ding, Z., Zhang, Q., Huang, X.: Automatic hashtag recommendation for microblogs using topic-specific translation model. Proceedings of COLING 2012: Posters pp. 265–274 (2012)

6. Godin, F., Slavkovikj, V., De Neve, W., Schrauwen, B., Van de Walle, R.: Using topic models for twitter hashtag recommendation. In: Proceedings of the 22nd International Conference on World Wide Web. pp. 593–596. ACM (2013)
7. Gong, Y., Zhang, Q., Han, X., Huang, X.: Phrase-based hashtag recommendation for microblog posts. Science China Information Sciences **60**(1), 012109 (2017)
8. Gong, Y., Zhang, Q.: Hashtag recommendation using attention-based convolutional neural network. In: IJCAI. pp. 2782–2788 (2016)
9. Graves, A., Wayne, G., Danihelka, I.: Neural Turing Machines. Arxiv pp. 1–26 (2014). https://doi.org/10.3389/neuro.12.006.2007, http://arxiv.org/abs/1410.5401
10. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation **9**(8), 1735–1780 (1997)
11. Huang, H., Zhang, Q., Gong, Y., Huang, X.: Hashtag recommendation using end-to-end memory networks with hierarchical attention. In: Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics:Technical Papers. pp. 943–952 (2016)
12. Kim, Y.: Convolutional neural networks for sentence classification. arXiv preprint arXiv:1408.5882 (2014)
13. Kowald, D., Pujari, S.C., Lex, E.: Temporal effects on hashtag reuse in twitter: A cognitive-inspired hashtag recommendation approach. In: Proceedings of the 26th International Conference on World Wide Web. pp. 1401–1410. International World Wide Web Conferences Steering Committee(2017)
14. Kywe, S., Hoang, T.A., Lim, E.P., Zhu, F.: On recommending hashtags in twitter networks. Social Informatics pp. 337–350 (2012)
15. Li, Y., Liu, T., Hu, J., Jiang, J.: Topical co-attention networks for hashtag recommendation on microblogs. Neurocomputing (2018)
16. Li, Y., Liu, T., Jiang, J., Zhang, L.: Hashtag recommendation with topical attention-based lstm. Coling (2016)
17. Liu, Z., Chen, X., Sun, M.: A simple word trigger method for social tag suggestion. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing. pp. 1577–1588. Association for Computational Linguistics (2011)
18. She, J., Chen, L.: Tomoha: Topic model-based hashtag recommendation on twitter. In: Proceedings of the 23rd International Conference on World Wide Web. pp. 371–372. ACM (2014)
19. Sukhbaatar, S., Weston, J., Fergus, R., et al.: End-to-end memory networks. In: Advances in neural information processing systems. pp. 2440–2448 (2015)
20. Tran, V.C., Hwang, D., Nguyen, N.T.: Hashtag recommendation approach based on content and user characteristics. Cybernetics and Systems pp. 1–16 (2018)
21. Wang, Y., Qu, J., Liu, J., Chen, J., Huang, Y.: What to tag your microblog: Hashtag recommendation based on topic analysis and collaborative filtering. In: Asia-Pacific Web Conference. pp. 610–618. Springer (2014)
22. Zhang, Q., Gong, Y., Sun, X., Huang, X.: Time-aware personalized hashtag recommendation on social media. In: Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers. pp. 203–212 (2014)
23. Zhao, F., Zhu, Y., Jin, H., Yang, L.T.: A personalized hashtag recommendation approach using lda-based topic model in microblog environment. Future Generation Computer Systems **65**, 196–206 (2016)