CTROF: A Collaborative Tweet Ranking Framework for Online Personalized Recommendation

Kaisong Song[1], Daling Wang[1,2], Shi Feng[1,2], Yifei Zhang[1,2], Wen Qu[1], and Ge Yu[1,2]

[1] School of Information Science and Engineering, Northeastern University
[2] Key Laboratory of Medical Image Computing (Northeastern University), Ministry of Education, Shenyang 110819, P.R. China
songkaisongabc@126.com,
{wangdaling,fengshi,zhangyifei,yuge}@ise.neu.edu.cn

Abstract. Current social media services like Twitter and Sina Weibo have become an indispensable platform, and provide a large number of real-time messages. However, users are often overwhelmed with large amounts of information delivered via their followees, and may miss out on much enjoyable or useful content. An information overload problem has troubled many users, especially those with many followees and thousands of tweets arriving every day. In this case, real-time personalized recommendation plays an extreme important role in microblog, which needs analyzing users' preference and recommending most relevant and newest content. Both of them pose serious challenges. In this paper, we focus on personal online tweet recommendation and propose a **C**ollaborative **T**weet **R**anking **O**nline **F**ramework (CTROF) for the recommendation, which has integrated the Optimized Collaborative Tweet Ranking model CTR+ and Reservoir Sampling algorithm together. The experiment conducted on a real dataset from Sina microblog shows good performance and our algorithm outperforms the other baseline methods.

Keywords: Bayesian Personalized Ranking, Latent Factor Model, Online Recommendation, Reservoir Sampling.

1 Introduction

In recent years, microblog such as Weibo and Twitter becomes very popular, because it allows users to post a short message named tweet or status for sharing viewpoints and acquiring knowledge in real time. According to statistics, more than 400 million tweets are generated per day in Twitter. As a result, the rich information in microblog not only expands our horizon, but also has wide applications in public opinions supervision, natural disaster prediction and political upheaval detection.

Microblog facilitates our life, but the information overload problem prevents it from developing further. A user usually follows many interested users such as friends, stars and organizations, and receives a lot of tweets at all times because of frequent updates. So the users are hard to consume so much content instantly in an effective way. In some cases, as for those with limited time to read, it's necessary to filter out those irrelevant and boring tweets by online recommending expected content.

V.S. Tseng et al. (Eds.): PAKDD 2014, Part II, LNAI 8444, pp. 1–12, 2014.

There are several challenges to be tackled for online personalized tweet recommendation. Firstly, although some latent factor models such as CTR [1] (Collaborative Tweet Ranking) have been proposed, yet they show poor performance as the dataset grows larger. Secondly, online algorithms can shorten the processing time, but they also reduce prediction quality. Thirdly, most online algorithms are generally based on recent data, and don't consider the history records which are relevant with users' customs and preferences. All the problems above have posed severe challenges for online tweets recommendation.

In this paper, we propose a Collaborative Tweet Ranking Online Framework (CTROF). Figure 1 below shows the algorithm process. Suppose we have some tweet history data in advance, we build an initial model called Optimized Collaborative Tweet Ranking model CTR+. For recommending the interested tweets in real time, we sample the tweet stream to update Double Reservoir for capturing the "sketch" of the stream. Then the training set is sampled and the initial CTR+ base model can be updated incrementally. Eventually, we get an online model for recommending interested tweets to users.

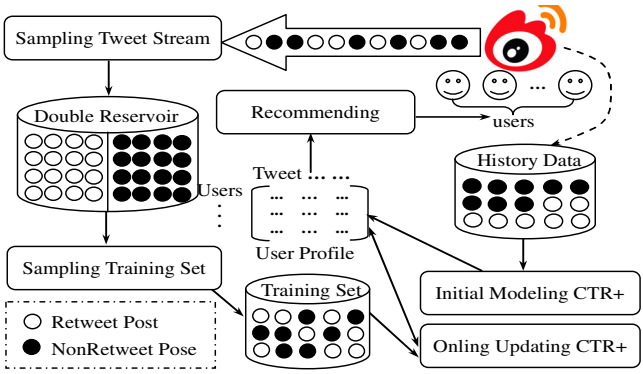

Fig. 1. Overview of CTROF in sampling and modeling online tweet stream

To the best of our knowledge, this work is the first experimental study to integrate tweet content (past and online), social structure and personal profile into collaborative filter model, and demonstrate a practical online personalized tweet recommendation framework. To summarize, the main contributions of our work are as follows.

(1) We propose a novel CTROF for online personal tweet recommendation. The novelty lies in a complete stream processing framework for real-time tweet ranking.

(2) We improve the performance of state-of-the-art tweet recommendation model CTR [1] by introducing personal hashtags to optimize BPR (Bayesian Personalized Ranking) [2], and creatively apply the model into online scenario.

(3) We use Reservoir Sampling algorithm for acquiring the "sketch" of incoming tweet stream dataset, which considers both the historical and the changing preferences.

(4) Finally, our algorithm considers the performance of time and space, and is able to balance the recommendation quality and the complexity of time and space well.

The structure of the rest of the paper is as follows. Related work is discussed briefly in Section 2. In Section 3, we briefly review CTR and propose a novel CTR+ recommendation model. In Section 4, we introduce the collaborative tweet ranking online framework CTROF. Section 5 introduces the classification of explicit features in detail. Section 6 compares CTROF with the other baseline models. In Section 7, we make a conclusion and point out the directions for our future work.

2 Related Work

CF (collaborative filtering) technique behind RS (recommender system) has been developed for years and kept to be a hotspot in academic and industrial field. Real applications include goods at Amazon, news at Google, movies at Yahoo and CDs at Netflix. In recent years, latent factor model proposed by Simon Funk has been widely applied in CF. Koren [3,4] and Xiang & Yang [5] improved it by considering neighborhood or time. Besides explicit feedback (rating), abundant implicit feedback [6] was also used. In high-order setting, Tensor Decomposition models [7,8] were studied. In addition, some learning methods were studied, such as Stochastic Gradient Descent (SGD), Alternating Least Squares [9] and Markov Chain Monte Carlo [10].

RS in microblog generally contains six categories factors in content: followee, follower, hashtag, tweet, retweet and URL. In recommendation pattern, it includes offline and online RS. In offline RS, [11] ranked incoming tweets by using author profile, syntactic feature, content and followee feature. Hong et al. [12] proposed a co-factorization machine to model interest and recommend relevant tweets. Chen et al. [1] proposed CTR model with high precision, however extra user preference signs, i.e. labels, were not considered. In addition, offline models are not suitable in online scenario with large continuous incoming data. In online RS, Diaz-Aviles et al. [13] proposed a RMFX online framework, however it had complex sampling algorithm and just considered hashtags of tweet. Work for ranking tweets also includes [14,15].

In this paper, we focus on recommending tweets in real time by integrating offline model and stream sampling algorithm together. We propose a novel CTROF framework by improving the drawbacks of Chen and Diaz-Aviles's work and absorbing their advantages. For this purpose, we first review CTR model briefly and propose an innovative CTR+ model by considering content, social relation and hashtags in Section 3. And then we utilize Reservoir Sampling [16] algorithm and propose a CTROF algorithm framework for ranking tweet stream in Section 4.

3 Optimized Collaborative Tweet Ranking Model

3.1 Notations Definition

We firstly define some notations being frequently used later. Let $U=\{u_1, u_2, \ldots, u_n\}$ be a user set, and $I=\{i_1, i_2, \ldots, i_m\}$ be a tweet set. Suppose there is interaction between any two entities, which shows the degree of interest. Then we get an interactive matrix $\mathbf{X}$: $U\times I$, and each element $x_{ui}\in\mathbf{X}$ represents an observation value. Predicting $\hat{x}_{ui}$ value can be seen as the task of estimating $\hat{\mathbf{X}}$. As we aim to get a personalized total ranking $>_u\subset I^2(?)$ of all tweets for a specific user u, we use Bayesian Personalized

Ranking for estimating $\hat{\mathbf{X}}$, instead of Root Mean Square Error (RMSE). As for any two entries $\hat{x}_{ui}$ and $\hat{x}_{uj}$ ($i \neq j$) in $\hat{\mathbf{X}}$, if $\hat{x}_{ui} > \hat{x}_{uj}$, then $i >_u j$.

After above modeling process, a basic offline model is built. Based on the model, we give out an optimized collaborative tweet ranking model CTR+ in offline scenario.

3.2 Optimizing Offline CTR to CTR+ Model

CTR [1] is an excellent offline RS, considers content, social relation, and explicit features simultaneously. In view of data sparsity and expandability, each tweet is decomposed into several words at topic level. All words from tweet set I constitute bag of words. Therefore, let $u \in U$ and $i \in I$, CTR model is described as follows:

$$\hat{x}_{u,i} = bias + p_u^T \left(\frac{1}{Z} \sum_{w \in W_i} q_w\right) \tag{1}$$

where W_i is word set of tweet i, q_w is vector of word $w \in W_i$, Σq_w is vector combination of tweet i, p_u is vector of user u, $bias = b_u + \Sigma b_w$ is bias term, b_u and b_w are user bias and word bias, and Z is normalization term which equals $|W_i|^{1/2}$ in general.

In addition, social relations are also important because users are more likely to retweet favorite publisher's tweets. As for any incoming tweet i, it can be mapped into corresponding publisher $p(i)$. So the formula can be further rewritten as:

$$\hat{x}_{u,i} = bias' + p_u^T \left(\frac{1}{Z} \sum_{w \in W_i} q_w + \kappa d_{p(i)}\right) \tag{2}$$

where $d_{p(i)}$ is publisher vector of i, $bias' = b_{p(i)} + bias$ is bias term, $b_{p(i)}$ is publisher bias and κ is an adjustable weighting parameter indicating publisher's importance relative to content.

In this paper, we introduce personal hashtags, a profile of personal interests and hobbies, into CTR model. Suppose users with similar interests are more likely to retweet each other. As for any tweet i, it can be mapped into its publisher's personal hashtag set $H_{p(i)}$. So we rewrite Formula (2) and represent CTR+ as:

$$\hat{x}_{u,i} = bias'' + p_u^T \left(\frac{1}{Z} \sum_{w \in W_i} q_w + \kappa d_{p(i)} + \frac{\beta}{Z'} \sum_{h \in H_{p(i)}} g_h\right) \tag{3}$$

where g_h is hashtag vector of any tag h in $H_{p(i)}$, $bias'' = bias' + \Sigma b_h$, b_h is tag bias, β is an adjustable weighting parameter indicating hashtags' importance relative to the content and $Z' = |H_{p(i)}|^{1/2}$ is normalization term.

Besides the above latent features, information such as tweet quality can also be incorporated into CTR+ as explicit features. Then $bias''$ term is replaced by $\sum br$, a weighted linear combination of $bias''$ and explicit feature biases. In the final CTR+ offline model, we get Formula (4) shown below:

$$\hat{x}_{u,i} = \sum_j b_j r_j + p_u^T \left(\frac{1}{Z} \sum_{w \in W_i} q_w + \kappa d_{p(i)} + \frac{\beta}{Z'} \sum_{h \in H_{p(i)}} g_h\right) \tag{4}$$

where b_j is any latent or explicit feature bias and r_j is weighting parameter represented by explicit feature value. For simplified formula, the weight r of b_u, b_w, $b_{p(i)}$ and b_h is set 1 by default. Details about explicit feature classification are discussed in Section 5.

Different from rating prediction, users just need to be recommended a list of sorted tweets. Similar to [1,12], retweet represents users' preference. Slightly different from Root Mean Square Mean in rating prediction, a BPR method is used instead.

Given a tweet set I, we should transform I into training set D in the form of tuples at first. For convenience, we define retweet set $R_u \subset I$ for any user u. Let $((u, i), (u, j)) \in D$ denotes a training instance, where $i \in R_u$ has been retweeted and $j \notin R_u$ not. Thus, D is formally defined as the tuple set from I, and we can describe it as: $D=\{((u, i), (u, j)) | i \in R_u \wedge j \notin R_u \wedge u \in U\}$. According to BPR Optimization Criterion, probability $p(\Theta)$ follows normal distribution $N(0, \Sigma_\Theta)$, in which diagonal matrix $\Sigma_\Theta=\lambda_\Theta E$, E is a unit diagonal matrix and λ_Θ is a constant, we aim to maximize the formula below:

$$\prod_{((u,i),(u,j)) \in D} \delta(\hat{x}_{uij}(\Theta)) \times p(\Theta) \tag{5}$$

where δ is sigmoid function. For convenience, Formula (5) is transferred as equivalent Formula (6) below by maximizing logarithm of posterior probability:

$$BPR - Opt := \max_{\Theta} \sum_{((u,i),(u,j)) \in D} \ln(1/1+e^{-(\hat{x}_{u,i}-\hat{x}_{u,j})}) - \lambda_\Theta \|\Theta\|^2 \tag{6}$$

In general, SGD is used for estimating parameter space Θ, and $\lambda_\Theta \|\Theta\|^2$ is a L2 regularization term. The training process of CTR+ model is shown below.

Algorithm. Training Offline Model CTR+ based on SGD for Θ estimation;

Input: Tweet training set D; Parameter space Θ; Relative weighting β and κ; Explicit feature weighting vector r; Latent factor number f; Learning rate η; Regularization parameters λ_Θ; Number of iterations T_Θ;

Output: Θ;

Description:

1) procedure CTR+Model(D, Θ, λ_Θ, f, η, T_Θ, β, κ, r);
2) initialize Θ;
3) for t=1 to T_Θ
4) for each p=((u,i), (u,j))$\in D$
5) $\Theta \leftarrow \Theta + \eta((e^{-\hat{x}_{uij}}/1+e^{-\hat{x}_{uij}}) \cdot (\partial \hat{x}_{uij} / \partial \Theta) - \lambda_\Theta \Theta)$;
6) return Θ;

4 Collaborative Tweet Ranking Online Framework

4.1 Building Online CTROF Model

In Section 3, we discussed CTR+ with tweet training set D. CTR+ is an offline model, because D is a static training set. As for new incoming tweet i^+, we calculate $\hat{x}_{ui^+}$ by decomposing i^+ into words, publisher and hashtag vectors. The larger $\hat{x}_{ui^+}$ is, the higher i^+ is ranked. Based on offline CTR+, we introduce CTROF in real-time scenario, which update model dynamically every time new tweets arrive.

In social network (such as Facebook) or microblogging service (like Twitter and Sina Weibo), messages are updated rapidly. A flow of messages constitutes data stream, called tweet stream in Twitter or Weibo. Diaz-Aviles [13] proved that Reservoir Sampling outperformed Single Pass, User Buffer, and captured the "sketch" of

history under the constraint of fixed space quite well. CTROF uses it and achieves online model by training CTR+ incrementally without retraining model completely.

Under the background of tweet stream, we use S to represent incoming tweet stream i_1, i_2… that arrives sequentially. As for tweet stream S, it is divided into retweet stream S_{ret} and non-retweet stream S_{nret}. Our algorithm maintains two fixed size Reservoirs R^+ and R^-, which contains random samples from S_{ret} and S_{nret}. So the key is to define reservoir $R^+=\{s_1, s_2,\ldots,s_{|R^+|}\}$ for S_{ret}, and reservoir R^- for S_{nret} as well. Similarly, let notation t^+ and t^- be tweet index for S_{ret} and S_{nret} respectively, reflecting the order of arrival of data in the stream. At the beginning, all incoming tweets will be pushed into reservoir R^+ and R^- continually and indiscriminately until $t^+=|R^+|$ and $t^-=|R^-|$. For subsequent t, we will decide whether a new incoming tweet will be put in reservoirs or not, and in which the old record will be replaced instead. The process of Collaborative Tweet Recommendation Online framework CTROF is shown below.

Algorithm. CTROF Framework;

Input: Tweet stream S; Reservoirs R^+ and R^-; Offline model parameters Θ'; Relative weighting β and κ; Explicit feature weighting vector r; Latent factor number f; Regularization parameters λ_Θ; Learning rate η; Number of iterations T_Θ for updateCTR+Model; Parameters c_{r} and c_{nr} control updates frequency of model;
Output: Θ;
Description:

```
1) procedure CTROF(S, Θ', R+, R−,λΘ, f, η, TΘ, β, κ, r, cr, cnr);
2) initialize Θ=Θ'; countr←0; countnr←0;
3) for t=1 to |S| do
4)   if t is retweet
5)     R+←ReservoirSampling(R+, it);
6)     countr←countr+1;
7)   else if t is non-retweet
8)     R−←ReservoirSampling (R−, it);
9)     countnr←countnr+1;
10)  if countr=cr and countnr=cnr
11)    Θ←updateCTR+Model (Θ,R+, R−, λΘ, f, η, TΘ, β, κ, r);
12)    countr←0, countnr←0;
13) Return Θ;
```

In above process, we selectively update two fixed-size reservoirs R^+ and R^- by Reservoir Sampling algorithm every time a new tweet arrives. For convenience, let R^* denotes R^+ or R^-. During R^* initialization, a new incoming tweet i_t is saved in the corresponding R^* directly until $t=|R^*|$. For subsequent t, random index μ is selected randomly within the scope of $|t|$. If $\mu\leq|R^*|$, we replace *t-th* tweet in R^* with tweet i_t. Above Reservoir Sampling ensures that each tweet is selected with equal probability.

4.2 Updating Online CTROF Model

In Algorithm CTROF Framework, updateCTR+Model (Line 11) updates the model incrementally by sampling training instances from R^*. We design a simple but effective sampling strategy by computing time distance between retweet and nonretweet. For formulization, as for any particular user u in each iteration, let pair (u, i) be

retweet selected randomly from reservoir R^+, and closest pair (u, j) from R^-, where distance $\delta=\min|Time_i-Time_j|$ and $1\le j\le|R^-|$. Then we select randomly m training instance pairs as *TrainSet*, and perform model update based on it.

Algorithm. Online Updating CTR+ based on Reservoir for Θ estimation;

Input: Tweet stream reservoirs R^+ and R^-; Relative weighting β and κ; Explicit feature weighting vector r; Latent factor number f; base model Θ'; Regularization parameters λ_Θ; Learning rate η; Number of iterations T_Θ;

Output: Θ;

Description:

1) procedure updateCTR+Model(Θ',R^+, R^-, λ_Θ, f, η, T_Θ, β, κ, r);
2) initialize $\Theta=\Theta'$; *TrainSet* ={ };
3) for t=1 to SampleNum
4) Draw pair (u, i) from R^+ randomly and closest negative (u, j) from R^-, save triple (u, i, j) into training set *TrainSet*;
5) for t=1 to T_Θ
6) for each triple (u, i, j) from *TrainSet*
7) $p_u \leftarrow p_u+\eta(\hat{e}(\frac{1}{Z^+}\sum_{w\in W_i} q_w^+ + \frac{1}{Z'^+}\sum_{h\in H_{p(i)}} g_h^+ - \frac{1}{Z^-}\sum_{w\in W_j} q_w^- - \frac{1}{Z'^-}\sum_{h\in H_{p(j)}} g_h^- + \kappa d_{p(i)}^+ - \kappa d_{p(j)}^-) - \lambda_u p_u)$;
8) $d_{p(i)}^+ \leftarrow d_{p(i)}^+ + \eta(\kappa\hat{e}p_u - \lambda_{p(i)} d_{p(i)}^+)$;
9) $d_{p(j)}^- \leftarrow d_{p(j)}^- - \eta(\kappa\hat{e}p_u + \lambda_{p(j)} d_{p(j)}^-)$;
10) for each $w\in W_i$ // W_i is the word set of tweet i
11) $q_w^+ \leftarrow q_w^+ + \eta(\hat{e}p_u / Z^+ - \lambda_w q_w^+)$;
12) for each $w\in W_j$ // W_j is the word set of tweet j
13) $q_w^- \leftarrow q_w^- - \eta(\hat{e}p_u / Z^- + \lambda_w q_w^-)$;
14) for each $h\in H_{p(i)}$ // $H_{p(i)}$ is the hashtag set of followee $p(i)$
15) $g_h^+ \leftarrow g_h^+ + \eta(\beta\hat{e}p_u / Z'^+ - \lambda_h g_h^+)$;
16) for each $h\in H_{p(j)}$ // $H_{p(j)}$ is the hashtag set of followee $p(j)$
17) $g_h^- \leftarrow g_h^- - \eta(\beta\hat{e}p_u / Z'^- + \lambda_h g_h^-)$;
18) for each explicit or latent feature bias k
19) $b_k \leftarrow b_k + \eta(\hat{e}(r_k^+ - r_k^-) - \lambda_k b_k)$;
20) return $\Theta=(p^*, d^*, q^*, g^*, b^*)$; // o^* represents any estimated parameter

Here notation o^+ denotes parameter vector of pair (u, i), while o^- for (u, j). We use $\hat{e}=(e^{-\hat{x}_{uij}}/1+e^{-\hat{x}_{uij}})$ for convenience. Given new reservoirs R^+ and R^-, we update model incrementally. Therefore, CTROF captures the history "sketch" and the current interest, and can overcome the problem of short-memory and avoids retraining model.

5 Relevant Features

In Section 3, we introduce CTR+ integrating linear combination of explicit features with *bias"* by bias term Σbr. In this section, we will further classify explicit features for capturing users' interests. Although [1,12] have defined different categories respectively, yet we will propose a more complete solution including four categories.

1) User Relationship Features: User relationship feature refers to the relationship between target user u and his/her friend v. It makes an assumption that: The more familiar with each other, the more likely to retweet his/her messages.

- **Co-Friends Score:** The similarity between u's followee set and v's.
- **Co-Follow Score:** The similarity between u's follower set and v's.
- **Mention Score:** The number of times u mentions v.
- **Retweet Score:** The number of times u retweets v.
- **Reply Score:** The number of times v replys to u.
- **Mutual Friend Score:** If u and v follow each other, it is 1, else 0.

2) Content Features: The features are the relevance between new incoming tweet i and the profiles of a target user u. Let $\mathbf{w}(i)$ as term set of i, $SP(u)$ as word set of u's status data, $RP(u)$ as word set of u's retweet data, $LP(u)$ as hashtag set of u's profile, $NP(u) = SP(u) \cup RP(u) \cup CP(u)$, and $\Re(\mathbf{w}_1, \mathbf{w}_2)$ as similarity between two term sets.

- **Relevance to Status:** $\Re(\mathbf{w}(i), SP(u))$ is the similarity between $\mathbf{w}(i)$ and $SP(u)$.
- **Relevance to Retweets:** $\Re(\mathbf{w}(i), RP(u))$ is similarity between $\mathbf{w}(i)$ and $RP(u)$.
- **Relevance to Hash Tags:** $\Re(\mathbf{w}(i), LP(u))$ is similarity between $\mathbf{w}(i)$ and $LP(u)$.
- **Relevance to Neighborhood:** $\Re(\mathbf{w}(i), NP(u))$ is similarity of $\mathbf{w}(i)$ and NP (u).

3) Tweet Features: Tweet features refer to the attributes of tweet, including general tweet length, hash tag count, URL count, reply count, retweet count. In addition, we add another two new features, that is, thumb up score and view score.

- **Thumb Up Score:** The number of times that tweet i is favorable or agreed.
- **View Score:** The number of times that tweet i is viewed.

4) Publisher Features: Publisher features represent the influence power of corresponding publisher of i, including not only mention, followee, follower and status count in [1], but also activity degree and loyalty degree. Let $time_{ui}$ be the time u publish tweet i, $\tau_u = \max\{time_{ui} - time_{uj}\} (i \neq j)$ as the period from first status to the last.

- **Loyal activity:** The feature measures how long the publisher u is active in RS. In general, we use τ_u to show the degree of loyalty.
- **Activity Degree:** The feature shows the activity of publisher and we may use $NT(u)/\tau_u$ to measure it, $NT(u)$ is the number of tweets that u has published.

6 Experiments

6.1 Experiment Setup

Our experiments are based on Sina Weibo platform and utilize the API tool [17]. Our work focuses on real-time personal tweet recommendation in Chinese microblog scenario and we use ICTCLAS [18] to handle word segmentation. For getting dataset, we randomly select a user and adopt user-based breadth-first traversal method by following followers and followees' links. Different from [1], our dataset includes tweet content, retweet action, personal hashtags and social relation. Retweeted and non-retweeted tweets are named positive and negative samples respectively. The dataset includes 46385 users' profile (user id, tags, followees) and their publishing historical data (tweet id, content, time, repost number). We select 675 users with more than 20 retweets, and others as their followees. Then tweets flood continuously from followees into corresponding followers in chronological order. Three fifths of dataset is as

training set and the others as testing set. Finally, training set for offline training contains 171,937 positive samples and 1,124,840 negative ones. Testing set contains 113,941 positive samples and 458,104 negative ones for offline testing. For testing stream dataset, we set parameters c_r and c_{nr} for controlling update frequency, and testing set can be further divided into tweet stream set $S=\{s_1, s_2 \ldots s_{11}\}$ by parameters, of which s_n ($1 \le n \le 10$) is for incremental online training, and s_{n+1} for online testing. The experiment shows that the crawled dataset coincides with our proposed model.

FM (Factorization Machines) [19] and SVD Feature [20] are generic factorization models tools. Considering coding workload and algorithm efficiency, we use the later.

Different from rating prediction, we focus on ranking tweets. So we measure recommendation precision by $P@N$ and recommendation quality by MAP metric. Let $MAP=\Sigma AP_u/|U|$ and $P@N=\Sigma p_u@n/|U|$. Given $u \in U$, $p_u@n$ is retweet proportion of top n in list, averaging precision (AP_u) is the average precision of each user:

$$AP_u = \frac{\sum_{n=1}^{N} p_u@n \times \delta(n)}{|R_u|} \tag{8}$$

where $\delta(n)$ is an indicator function, which returns 1 if *n-th* tweet in the list is retweeted, and 0 otherwise. $|R_u|$ is total number of retweeted tweets in top N list, and $|R_u| \le N$. And $p_u@n$ measures the precision of top n tweets.

6.2 Experiment Result

In Section 3, we have proposed CTR+ model for offline tweet recommendation system modeling. CTR+ includes necessary components (explicit factor, term factor, social factor and hash tag factor). For studying components' influence, we make a comparison by $s_1 \in S$ in Figure 2. We compare MAP by N=15 and $P@N$ by setting N to 5, 10 and 15. The number of iterations and factors is set to 40 and 64 respectively. Relative weight parameters β and κ are set 0.8 uniformly. CTR performs well (MAP=0.8074~0.8114) when β is around 0.8, so we choose best parameter 0.8. Given fixed β, we randomly select κ=0.8 because MAP remains stable when κ ranges from 0.7 to 1. As large training dataset rarely encounter the over-fitting problem, the regularization parameter λ is set 0.005. For approaching optimal value, the learning rate η is set small value 0.004, despite a certain loss in convergence rate.

Figure 2 shows the precision of CTR+ is always higher than CTR, and reflects the importance of single component and their combination. Chronological method's precision is shown for reference. For simplicity, we just choose explicit features (Text Length, Retweet Score, and Relevance to Hash Tags) as global features. We find that explicit features, term, hash tag, and social component improve MAP by 54%, 70%, 86% and 92% respectively relative to chronological method, which indicates all components are necessary and effective. CTR contains all components except hash tag, and outperforms any single component. However, CTR+, compared with CTR, improves precision by 12.3%, which indicates that our model is better.

Figure 3 shows that runtime convergence of different models. All models have different convergence rate and converge to steady values after 30 rounds. So our offline base model is reasonably set to 40 rounds. In addition, we calculate $P@5$ value by

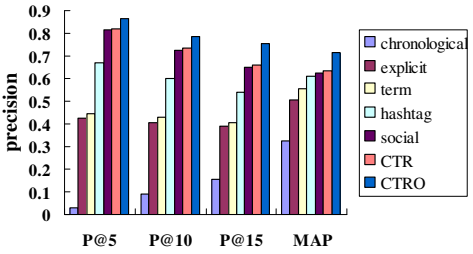

Fig. 2. Evaluation of Compared Methods

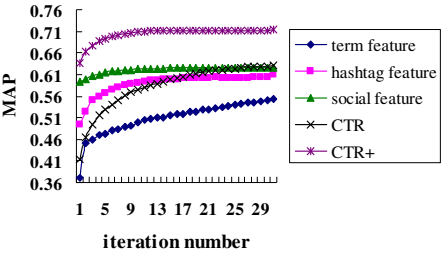

Fig.3. Runtime Convergence of CTR+

setting factor number to 32, 64, 80, 96, and 112. Term Feature (0.4438~0.4439), Tag Feature (0.6644~0.6752), Social Feature (0.8114~0.8124), CTR (0.8219~0.8229) and CTR+ (0.8637~0.8641) remains stable. Therefore, we set 64 factors reasonably.

We imitate tweets flow into our framework in chronological sequence continually. In addition, we let each one' tweet stream arrive at the same opportunity. Retweetes in testing set are far less than non-retweets, and the ratio is about 1/4. So we set the size of reservoirs R^+ and R^- to 10,000 and 40,000 respectively for reflecting real data distribution. Therefore, we won't update our base model until 10,000 retweets and 40,000 non-retweets arrive. In order to verify prediction precision of top-N items in recommendation list, N is set to 5, 10 and 15 respectively. We compare our stream framework CTROF with two offline models CTR and CTR+. The MAP for each method, in different list size, is shown in Figure 4 below.

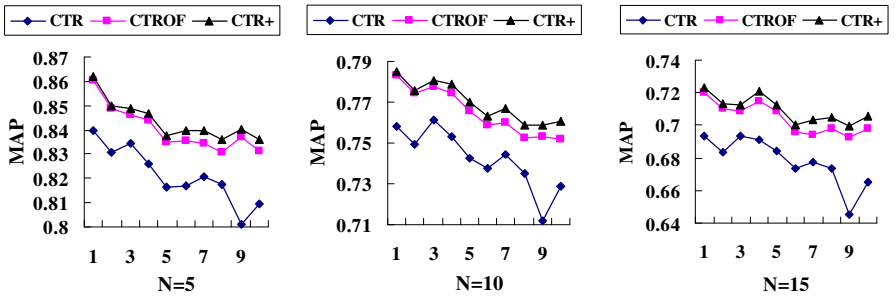

Fig. 4. Recommendation precision of different sized list. The number of factor is 64 for CTR, CTR+, and CTROF. S_i ($1 \le i \le 10$) denotes the incoming retweet and nonretweet tweet stream by setting c_r=10,000 and c_{nr}=40,000, which is also as big as the size of reservoir R^+ and R^-.

Figure 4 shows that online model CTROF achieves better performance over offline CTR, and slightly below than offline CTR+ model. Compared with CTR+, CTROF just capture information sketch by sampling, so precision is slightly lower. In addition, additional hash tag factor represents personal preference and makes CTROF outperform CTR model. As online recommendations focus more on comprehensive performance of runtime, space and precision, our method saves lots of runtime and space and precision is close to best offline model CTR+.

Next, we further discuss time, space and recommendation precision comparison of CTROF and CTR+, which are implemented by SVDFeature tool in C++. We ran

CTR+ and CTROF on an Intel Core i7-2600 3.4GHz CPU and 2G memory virtual machine with Linux 32bit operating system. As none of the methods is parallel, we run the program on a single CPU. As the platform and implementation technique influence the performance greatly, so the setting can be used as a reference indicator.

The performance of CTROF is not only related to the number of factors and iterations, but also related to the training set size of the initial base model and the reservoir size. If we train base model with a very big training set about 1.3 million tweets, then the size of reservoir has little impact on the recommendation quality. That's because long-term accumulated dataset may include almost all possible situations in terms, retweet relation and tags, so a small amount of tweet stream will not change the model greatly. So we just choose about 0.4 million training instances, and set different sized reservoirs. The experiment comparison result is shown below.

Table 1. Comparison among time, space, and recommendation quality with different sized reservoir R^+ and R^-. The number of iteration and factors is set to 40 and 64. The base model CTR+ is trained by 0.4 million tweets. List length N is 40. The testing set has 0.1 million tweets. Time is the runtime sum of six tests, and Space is the disk space of training text.

Method (64 factors, 40 r) Reservoir size of R^+, R^-	Time (second)	Space	MAP	Recommendation Quality of CTROF
CTR+ [Baseline]	464s	100%	0.802	100%
CTROF R^+=1,000 R^-=4,000	40s	1.11%	0.752	93.82%
CTROF R^+=5,000 R^-=20,000	66s	5.25%	0.771	96.13%
CTROF R^+=10,000 R^-=40,000	92s	10.43%	0.780	97.23%

Table 1 shows that the reservoir size can influence the final recommendation quality obviously. Offline model CTR+ is used as a reference. When reservoir is big enough, the data distribution is much more appropriate and close to real data distribution. In this paper, we set reservoir size to T_1=(R^+=1,000: R^-=4,000), T_2=(R^+=5,000: R^-=20,000), and T_3=(R^+=10,000: R^-=40,000) respectively. By comparison, we find T_2 is close to T_3 in recommendation quality, but faster than T_3 by 28.2%. In addition, T_2 outperforms T_1 by 2.5% within the scope of tolerable time. So we can draw a conclusion that recommendation quality is good enough when $5{,}000 \leq |R^+| \leq 10{,}000$ and $R^- = 4|R^+|$.

7 Conclusions and Future Work

In this paper, we propose an offline ranking model CTR+ which considers explicit feature, content, social relation and personal hashtags. Moreover, we propose a novel tweet ranking online framework CTROF for real-time personalized recommendation. CTROF integrates Reservoir Sampling algorithm and CTR+ together, which captures "sketch" of tweet historical data, and absorbs new preference change from incoming tweet stream in the meantime. By experiments, we show that CTR+ outperforms CTR offline model and CTROF can capture real data distribution and achieve quite good precision, which demonstrates that our proposed method is effective and efficient.

Future work includes further analyzing semantics of content and studying more accurate and efficient sampling methods for improving the recommendation quality. We will consider more media factors in tweet such as images and videos. In addition, the tensor factorization for tweet recommendation is also our future direction.

Acknowledgements. This work is supported by the State Key Development Program for Basic Research of China (Grant No. 2011CB302200-G), State Key Program of National Natural Science of China (Grant No. 61033007), National Natural Science Foundation of China (Grant No. 61100026, 61370074), and Fundamental Research Funds for the Central Universities (N100704001, N120404007, N100304004).

References

1. Chen, K., Chen, T., Zheng, G., Jin, O., Yao, E., Yu, Y.: Collaborative personalized tweet recommendation. SIGIR, 661–670 (2012)
2. Rendle, S., Freudenthaler, C., Gantner, Z., Schmidt-Thieme, L.: BPR: Bayesian Personalized Ranking from Implicit Feedback. CoRR abs/1205.2618 (2012)
3. Koren, Y.: Factorization meets the neighborhood: A multifaceted collaborative filtering model. In: KDD 2008, pp. 426–434 (2008)
4. Koren, Y.: Collaborative filtering with temporal dynamics. In: KDD 2009, pp. 447–456 (2009)
5. Xiang, L., Yang, Q.: Time-Dependent Models in Collaborative Filtering Based Recommender System. In: Web Intelligence, pp. 450–457 (2009)
6. Oard, D.W., Kim, J.: Implicit Feedback for Recommender Systems. In: Proc. 5th DELOS Workshop on Filtering and Collaborative Filtering, pp. 31–36 (1998)
7. Rendle, S., Marinho, L.B., Nanopoulos, A., Schmidt-Thieme, L.: Learning optimal ranking with tensor factorization for tag recommendation. In: KDD 2009, pp. 727–736 (2009)
8. Symeonidis, P., Nanopoulos, A., Manolopoulos, Y.: Tag recommendations based on tensor dimensionality reduction. In: RecSys 2008, pp. 43–50 (2008)
9. Pilászy, I., Zibriczky, D., Tikk, D.: Fast als-based matrix factorization for explicit and implicit feedback datasets. In: RecSys 2010, pp. 71–78 (2010)
10. Salakhutdinov, R., Mnih, A.: Bayesian probabilistic matrix factorization using Markov chain Monte Carlo. In: ICML 2008, pp. 880–887 (2008)
11. Uysal, I., Croft, W.B.: User oriented tweet ranking: A filtering approach to microblogs. In: CIKM 2011, pp. 2261–2264 (2011)
12. Hong, L., Doumith, A.S., Davison, B.D.: Co-factorization machines: Modeling user interests and predicting individual decisions in Twitter. In: WSDM 2013, pp. 557–566 (2013)
13. Diaz-Aviles, E., Drumond, L., Schmidt-Thieme, L., Nejdl, W.: Real-time top-n recommendation in social streams. In: RecSys 2012, pp. 59–66 (2012)
14. Feng, W., Wang, J.: Retweet or not?: Personalized tweet re-ranking. In: WSDM 2013, pp. 577–586 (2013)
15. Hong, L., Bekkerman, R., Adler, J., Davison, B.: Learning to rank social update streams. In: SIGIR 2012, pp. 651–660 (2012)
16. Vitter, J.S.: Random Sampling with a Reservoir. ACM TOMS 11(1), 37–57 (1985)
17. `http://open.weibo.com/`
18. `http://ictclas.nlpir.org/`
19. Rendle, S.: Factorization Machines with libFM. ACM TIST 3(3), 57 (2012)
20. Chen, T., Zhang, W., Lu, Q., Chen, K., Zheng, Z., Yu, Y.: SVDFeature: A Toolkit for Feature-based Collaborative Filtering. JMLR 13(Dec), 3619–3622