Collaborative Personalized Twitter Search with Topic-Language Models

Jan Vosecky, Kenneth Wai-Ting Leung, Wilfred Ng
Department of Computer Science and Engineering
Hong Kong University of Science and Technology
Clear Water Bay, Kowloon, Hong Kong
{jvosecky,kwtleung,wilfred}@cse.ust.hk

ABSTRACT

The vast amount of real-time and social content in microblogs results in an information overload for users when searching microblog data. Given the user's search query, delivering content that is relevant to her interests is a challenging problem. Traditional methods for personalized Web search are insufficient in the microblog domain, because of the diversity of topics, sparseness of user data and the highly social nature. In particular, social interactions between users need to be considered, in order to accurately model user's interests, alleviate data sparseness and tackle the cold-start problem. In this paper, we therefore propose a novel framework for Collaborative Personalized Twitter Search. At its core, we develop a collaborative user model, which exploits the user's social connections in order to obtain a comprehensive account of her preferences. We then propose a novel user model structure to manage the topical diversity in Twitter and to enable semantic-aware query disambiguation. Our framework integrates a variety of information about the user's preferences in a principled manner. A thorough evaluation is conducted using two personalized Twitter search query logs, demonstrating a superior ranking performance of our framework compared with state-of-the-art baselines.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Retrieval models*

Keywords

Collaborative personalized search; Twitter; topic modeling; language modeling

1. INTRODUCTION

In recent years, microblogging services, such as Twitter, emerged as a popular platform for real-time information exchange. Every day, nearly 60 million short messages (*tweets*) are published and over 2 billion search queries are issued in Twitter[1]. However, the vast amount of content in Twitter results in an information overload for users when searching microblog data. In particular, tweets cover a wide range of topics and purposes, which makes a user's search for relevant content challenging and time-consuming. As a result, novel methods for search result personalization are needed in the microblogging domain.

Although much work has been done on personalized Web search [5, 12, 18, 19, 25] and collaborative Web search [17, 21, 23, 27], little work has been done on personalizing the search experience in the social environment of Twitter. Recent work related to information retrieval in Twitter does not consider individual users' interests in the ranking [7, 8, 14, 16]. Thus, in this paper we develop an effective framework for collaborative personalized Twitter search.

Similarly to personalized Web search, our goal is to rerank a set of search results based on their similarity with the user's preferences and thus, improve the retrieval effectiveness. However, the microblogging environment differs from traditional Web significantly, which calls for novel methods to accurately model user's preferences. The main challenges related to personalized information retrieval in microblogs can be summarized as follows:

- *Highly social.* Each user can be seen both as a content producer and consumer, and have rich interactions with other users [11]. Utilizing this social environment to model user's preferences is not trivial and requires a careful selection of relevant social content.
- *Diversity of topics and purposes.* Content in microblogs covers very diverse topics and purposes [2, 11]. Failure to distinguish the various types of information may result in noisy and inaccurate user models.
- *Data sparseness.* Effective user modeling methods need to tackle the sparseness of user's data, such as the short length and limited amount of user's tweets, few interactions with other users, or a limited search history.
- *Dynamic and real-time.* The high volume of microblogs calls for models that are rapidly updatable, adapt to the constantly evolving semantics in microblogs and reflect updates in the social network.

In this paper, we address the above challenges and propose a novel probabilistic framework for *Collaborative Personalized Twitter Search (CPTS)*. In the following paragraphs, we highlight the main features of our framework.

Collaborative User Modeling. The user's social connections can provide valuable clues about her preferences. However, constructing a collaborative user model in not trivial, since not all information from the user's social environment is equally useful. In fact, the collaborative model may

[1] http://www.statisticbrain.com/twitter-statistics/

SIGIR'14, July 6–11, 2014, Gold Coast, Queensland, Australia.
Copyright is held by the owner/author(s). Publication rights licensed to ACM.
ACM 978-1-4503-2257-7/14/07 ...$15.00.
http://dx.doi.org/10.1145/2600428.2609584.

become noisy if all information from the user's "friends" is included. Therefore, we analyze the user's social interactions and estimate the importance of each "friend". Furthermore, we analyze the importance of each friend's topic, in order to separate potentially relevant topics from irrelevant ones. The proposed *collaborative user model* helps to tackle the sparseness of individual user's data while avoiding the injection of unnecessary noise from the social neighborhood. Moreover, this method is suitable to new users who have posted few tweets, thus addressing the *cold-start* problem.

Topic-Specific Language Modeling. Our approach to user modeling and personalized re-ranking is based on *topics*. Content posted by a user may be highly diverse in terms of topics (e.g., "business", "sport", but also "emotional comments"). Putting all information from a user into a single user model would lead to a noisy and inaccurate model. Therefore, we distinguish the different kinds of information and propose a novel user model structure, referred to as *topic-specific user language models*. The proposed structure is beneficial in several ways. First, it enables effective *query disambiguation* by estimating the latent meaning behind a user's query. Second, during personalized re-ranking, we may identify tweets from relevant topics and promote them in the ranking. Third, we consider the user's topical preferences when building the collaborative user model.

Integrated Posting-Search Model. Each microblog user is both a content producer and consumer. On the one hand, a user's tweets indicate her preferences as a content producer. On the other hand, user's search activity indicates her preferences as an information consumer. Our framework integrates both types of preferences in a principled manner.

Responsive and Dynamic Profiles. Our user models are dynamically updatable, with adjustable weights of each component. Furthermore, our framework does not require any explicit input from the users to maintain their profiles.

We summarize contributions in this paper as follows:

- We propose a novel framework for Collaborative Personalized Twitter Search (CPTS). The framework integrates user's preferences, social environment and search history in a principled manner. The obtained user models provide comprehensive evidence for query disambiguation and search result re-ranking.
- We develop a novel collaborative user modeling method, based on the analysis of user's interactions and topical preferences. The model is built with detailed parameterization of the influence of each friend and each topic. Our evaluation shows that the collaborative model significantly improves ranking effectiveness.
- We propose a user model structure, referred to as topic-specific user language models. The method enables better organization of user preferences, topic-specific analysis of user interactions and semantic-aware reranking of search results. Our evaluation shows that the proposed user model structure consistently outperforms state-of-the-art baselines for search personalization.
- We present a comprehensive experimental evaluation of our framework using two personalized search query logs and compare the ranking performance against multiple state-of-the-art baselines. Our evaluation shows that our framework produces significant ranking improvements over the baseline methods.

The rest of this paper is organized as follows. We review related work in Section 2 and preliminaries in Section 3. Our proposed framework is presented in Section 4. Section 5 presents our evaluations. Section 6 concludes our findings.

2. RELATED WORK

Personalized Web Search. In personalized Web search [5, 12, 18, 19, 25], the principle of search result re-ranking is usually applied. Given a set of search results to a user's query, we promote search results that have a higher similarity with the user's preferences (represented as the *user model*), in addition to traditional user-independent metrics used in the ranking, such as the query-document relevance and document-specific features. Building the user model in mostly relies on implicit data from user's clicks. However, only considering the information from single user's clicks results in the problems of *data sparseness* and *cold-start* [27].

Collaborative Web Search. To alleviate the data sparseness problem in personalized Web search, collaborative Web search techniques were developed [17, 21, 23, 27]. In collaborative Web search, the search preferences of a community of users are mined and utilized in a similar way to collaborative filtering. Search results are then re-ranked for a given user based on the pages clicked by other similar users. For example, CubeSVD [21] analyzes the correlation between users, queries and documents in a search query log. The extracted click patterns among a community of users are then employed for personalizing the results of a particular user. Xue et al. [27] take a language modeling approach to build user-specific language models and cluster similar users into communities. A community-specific language model is then used for smoothing the user models to inject community knowledge. In contrast, our approach exploits the explicit social neighborhood of a user to learn about her information need. We measure the importance of each social connection and construct a topic-sensitive collaborative user model.

Microblog Search. In terms of general information retrieval in Twitter, Massoudi et al. [14] presents a retrieval model for microblogs, which takes into account tweet-query relevance, quality features of tweets and incorporates a query expansion model. Duan et al. [7] use a learning-to-rank approach for general tweet ranking. [8, 15] incorporate temporal aspects of tweets to improve microblog search.

Some attempts were made to construct a user profile from microblog data for the purpose of recommendation [1, 3, 4]. For example, Chen et al. [4] take a collaborative ranking approach to tweet recommendation and employs a number of tweet-specific features to influence the importance of a tweet. However, the existing work does not address the diversity of topics or user's social connections when constructing the user model. Moreover, the user's query has not been considered and thus the methods cannot be readily applied to microblog search personalization.

To the best of our knowledge, our work is the first to establish a collaborative Twitter-based search personalization framework and present an effective means to integrate language modeling, topic modeling and social media-specific components into a unified framework.

3. PRELIMINARIES

3.1 Language Modeling IR

Statistical language modeling (LM) has been successfully applied in machine translation, speech recognition and information retrieval [28]. In information retrieval, LM typically adopts the Query Likelihood model [28]: Given a query

$Q = \{q_1, \dots, q_i\}$ and a document $D = \{w_1, \dots, w_j\}$, the score for D against Q is proportional to the probability that the multinomial language model that generated D also generated Q. Formally,

$$P_{LM}(D|Q) \propto P(Q|D)P(D). \quad (1)$$

This ranking method captures both the document's relevance to the query and also the document's prior probability, $P(D)$. The latter may be used to incorporate any document-specific features, such as PageRank.

Assuming the unigram model of documents, we can decompose the query into individual words and compute the overall score as a product of individual term scores,

$$P(Q|D) = \prod_{w \in Q} P(w|D). \quad (2)$$

An important step in the estimation of the conditional probability $P(w|D)$ is to account for unobserved terms. To this end, several smoothing methods were proposed. Jelinek-Mercer smoothing [28] is one of the simplest and most popular smoothing methods that uses a fixed smoothing parameter λ to interpolate a document's language model with a global (corpus) model. It is defined as

$$P(w|D) = (1 - \lambda)\frac{c(w, D)}{|D|} + \lambda p(w|C), \quad (3)$$

where $c(w, D)$ is the count of word w in document D and $p(w|C)$ is the probability of w in the entire corpus.

3.2 Topic Modeling and IR

Topic modeling (TM) has gained popularity in recent years as a tool to perform unsupervised analysis of text collections and organize documents by their latent topics. One of the most popular topic models is Latent Dirichlet Allocation (LDA) [9]. LDA can be used to discover a set of K latent topics from a document corpus, and then to represent each document D as a mixture θ_D of the latent topics. For each word w_i in D, we first sample a topic z_i from the document mixture θ_D. Second, we sample w_i according to topic z's word distribution ϕ_z.

Much work has been done on developing efficient inference methods for LDA. Recently, online inference for LDA has been developed in [9], which enables LDA to be trained on massive and streaming data. We use the algorithm in [9] to train LDA on a large Twitter dataset in an online fashion.

Topic modeling has previously been applied for information retrieval [26]. In topic model-driven IR, the probability of a query given document is

$$P_{TM}(Q|D) = \sum_{z=1}^{K} P_{TM}(Q|\phi_z)P(\theta_{D,z}). \quad (4)$$

However, this approach often resulted in decreased ranking accuracy compared with standard LM, since topics are too coarse [26]. To alleviate this problem, a linear combination of TM and document LM is usually employed,

$$P(Q|D) = \lambda P_{TM}(Q|D) + (1 - \lambda)P_{LM}(Q|D). \quad (5)$$

In face of the short length and diverse topics of tweets, neither LM nor TM alone are suitable to build user models for personalized microblog search. On the one hand, simply employing LM and estimating user-specific language models (e.g., in [27]) may easily promote irrelevant tweets in the ranking. Using such a model, even the match of a few words of an irrelevant tweet with the user model may boost its ranking. On the other hand, a pure topic modeling approach to represent user's preferences may yield even worse results, since the topics are too coarse [26]. Matching a tweet to topic "IT" may not guarantee its relevance to the user's preferences. Thus, we develop a novel user model structure, which enables a fine-grained and topic-aware user representation.

4. PROPOSED FRAMEWORK

First, we define the scope of our personalization framework. *Given a microblog user u, a search query Q and a set of N microblog documents returned by a base search engine, our goal is to re-rank the documents using query Q and a user model M_u, such that documents matching the user's interests are ranked at top positions.*

To achieve this goal, we need to solve two basic problems: (1) how to construct the user model M_u, and (2) how to utilize this model for document ranking. In this section, we present our personalization framework to meet the above goals. We note that throughout this paper, a *document* refers to a single microblog message (i.e., a *tweet*).

We begin our discussion with some basic assumptions made in our framework. In particular, we recognize the importance of analyzing user preferences in microblogs in terms of *topics*. Our observation is that microblog content is highly diverse in terms of topics and purposes. This diversity is discussed in detail in [11, 20]. For example, Java et al. [11] found that Twitter serves a wide range of purposes, such as daily chatter, conversations or news sharing. In this paper, we simply use the concept of *topics* to broadly refer to the different kinds of content. An example of such topics would be "pop music", "IT news", but also "personal feelings". We note that even the interests of a single user may be very diverse. As a result, our intuition is that by treating all information with the same importance, we would obtain an inaccurate and noisy personalization model. Therefore, we propose to distinguish the different kinds of information within our framework.

State-of-the-art topic models such as LDA [9] may be employed for unsupervised topic discovery and for topic assignment of future documents. As the first step, we therefore build a global *Topic Model* using a large Twitter corpus, which will be utilized throughout our framework. We will refer to this model simply as TM.

We now define some basic operations done using the TM. To obtain a topic distribution θ_D of a new document D, we obtain each dimension i of θ_D as follows

$$\theta_{D,i} = \frac{\prod_{w \in D} P(w|\phi_i)}{\prod_{w \in D} \sum_k P(w|\phi_k)}. \quad (6)$$

To assign document D to a single topic, we choose the topic that maximizes the probability of generating D,

$$z_D = \arg\max_k \prod_{w \in D} P(w|\phi_k). \quad (7)$$

4.1 Modeling an Individual User

In contrast to previous approaches, which estimate a single language model for each user (e.g., in [27]), our approach is to construct a two-layer user model. Each user model is composed of a topic layer and a word layer. The topic layer represents user's high-level preferences and the word layer represents the user's words used within the respective topic. We refer to this model the *Individual* User *Model* (IM).

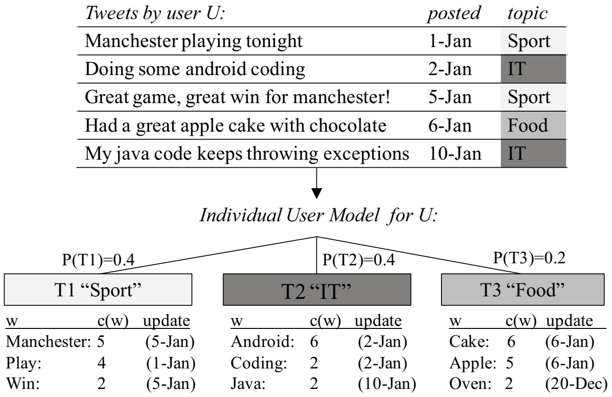


Figure 1: Individual User Model

The two-layer structure has the advantage of organizing user preferences related to different topics separately. This in turn enables semantic-aware query disambiguation and search result re-ranking. For example, Figure 1 illustrates the IM of user U. U often tweets about IT and mentions the term "android". Also, U tweets about food and mentions the term "apple". Thus, if U searches for "android", U's IM suggests that U may be interested in IT-related tweets. Also, if U issues another query related to IT (e.g., "mobile apps"), tweets mentioning "android" will be ranked high. In contrast, using a traditional single-layer user model would also falsely promote tweets mentioning "apple".

We estimate the IM for each user u in the following way. First, we assign each microblog document D (i.e., a tweet) from u to a topic using Equation (7). Second, we build a language model for each u's topic using all u's documents assigned to the respective topic. On the word level, the maximum likelihood (ML) estimate of the probability of word w in topic k for user u is defined as

$$\theta^{IM}_{u,k,w} = \frac{\sum_{D:D\in\mathbf{D}_u\wedge z_D=k} c(w,D)}{\sum_{w'\in V}\sum_{D:D\in\mathbf{D}_u\wedge z_D=k} c(w',D)}. \tag{8}$$

where $\mathbf{D}_u$ is the set of documents by user u, $c(w,D)$ is the count of word w in D, V is the vocabulary, z_D is the topic of D. We refer to this probability as $\theta^{IM}_{u,k,w}$ for short.

On the topic level, the probability that u chooses topic k is estimated as

$$\theta^{IM}_{u,k} = \frac{|\{D : D \in \mathbf{D}_u \wedge z_D = k\}|}{|\mathbf{D}_u|}. \tag{9}$$

Figure 1 illustrates an IM created from a set of user's documents.

Before the IM can be used by a ranking function to get the user's preference for word w in topic k (i.e., $\theta^{IM}_{u,k,w}$), we need to account for the case of unobserved words. To this end, we smooth the topic-word distribution in the IM using the underlying topic model,

$$\theta^{I\hat{M}}_{u,k,w} = (1-\lambda)\theta^{IM}_{u,k,w} + \lambda P(w|\phi^{TM}_k), \tag{10}$$

where λ is a parameter for Jelinek-Mercer smoothing.

If we further incorporate the topic-level probabilities of user u, we get

$$\theta^{I\hat{M}}_{u,k,w} = (1-\lambda)\theta^{IM}_{u,k,w}\theta^{IM}_{u,k} + \lambda P(w|\phi^{TM}_k)\eta, \tag{11}$$

where η is the prior probability of choosing a topic. In our work, we choose a constant value for η.

Additionally, along with each $\theta^{IM}_{u,k,w}$, we also store the timestamp of the latest document in topic k containing w. This allows to track the recency of information in the IM. The probability of w in the IM can then be re-defined as

$$\theta^{IM}_{u,k,w} \propto \theta^{IM}_{u,k,w} \cdot e^{-\rho\cdot t_w}, \tag{12}$$

where ρ is the forgetting coefficient. This time decay factor assumes an exponential forgetting rate and was applied to IR by Li and Croft [13].

4.2 Basic Personalization Model

Based on the individual user model defined above, we formulate a basic personalized ranking function as follows:

$$P(D,Q,u) \propto \Big(\sum_{k=1}^{K} P(Q|\hat{\theta}^{IM}_{u,k,w})P(D|\hat{\theta}^{IM}_{u,k,w})\Big)P(D), \tag{13}$$

where $P(Q|\hat{\theta}^{IM}_{u,k,w})$ and $P(D|\hat{\theta}^{IM}_{u,k,w})$ are topic-specific personalized scores of D and Q, respectively, and $P(D)$ is the document prior.

This approach essentially decomposes the ranking into two components. First, we perform *query disambiguation* using u's IM. That is, we predict which underlying topic the user had in mind when formulating the query. Second, the obtained probability given topic k is multiplied with the probability that document D belongs to the respective topic in u's IM.

To obtain $P(Q|\hat{\theta}^{IM}_{u,k,w})$ (and similarly, $P(D|\hat{\theta}^{IM}_{u,k,w})$), we compute the product of the scores of each word,

$$P(Q|\hat{\theta}^{IM}_{u,k,w}) = \prod_{w\in Q} P(w|\hat{\theta}^{IM}_{u,k,w}). \tag{14}$$

4.3 Collaborative User Modeling

One of the most important features of microblogs is its social network structure, which enables interactions between users. Users may follow other users, such as public figures or their real-world friends, and are able to receive their tweets. If a user finds a tweet interesting, they are able to re-tweet it or add it to their favorites. Furthermore, users can have conversations or mention each other in their tweets. This social environment presents rich additional information about the user's interests, which can increase the completeness of the user model and tackle data sparseness of an individual user. In this work, our main focus is on the followees of a user (i.e., the users one has subscribed to), which we refer to as *friends* for simplicity.

However, we observe that different friends may have a different influence on a particular user u. For example, u may follow hundreds of friends, but only frequently interacts with a small fraction of them. Furthermore, not all content posted by a friend may be of interest to u. For example, u may be interested in tweets from friend f about "IT news", but may not be interested in f's comments about "relationships".

We therefore assign a weight to each friend of user u, which is composed of four factors:

Popularity weight $w_P(f)$**:** The popularity of user f, which may be indicated by f's number of followees, number of times f is listed in public lists, PageRank score, etc. In our work, the log of the followee count is used as an indicator of popularity. The popularity weight is normalized by $w_P(f) = \log(popularity)/\log(max)$, where max is the maximum popularity of a user in Twitter[2].

Affinity $w_A(u,f)$**:** We measure the similarity of interests of u and f as the inverse KL-divergence between their topic-level profiles, $w_A(u,f) = 1/KL(\theta^{IM}_{u,\circ}||\theta^{IM}_{f,\circ})$.

Topic-interaction weight $w_I(u,f,k)$**:** We analyze the interactions between u and f, which include the conversations between u and f, mentions of f by u, and re-tweets of f's tweets by u. We first retrieve all tweets containing the above interactions and assign each tweet to a single topic (for conversations, we assign the entire conversation to a topic). The topic-interaction weight $w_I(u,f,k)$ is then based on the count of interactions between u and f that are assigned to topic k, denoted $c(u,f,k)$. The weight is normalized by $w_I(u,f,k) = \log_{10}(1 + c(u,f,k))$ if $c < \iota$ and $w_I(u,f,k) = 1$ if $c \geq \iota$. We set ι empirically to $\iota = 10$.

Topic bias $w_T(u,k)$: Apart from the above friend-dependent weights, we also consider u's bias towards content about topic k, i.e. $w_T(u,k) = \theta^{IM}_{u,k}$. If f's IM contains topic k, we apply the topic bias as a prior probability of u's interest.

The overall weight of friend f is then a vector $\omega_{u,f}$, where each dimension $k \in \{1, \ldots, K\}$ is defined as

$$\omega_{u,f,k} = \sigma^T . \begin{pmatrix} w_P(f) \\ w_I(u,f,k) \\ w_A(u,f) \\ w_T(u,k) \end{pmatrix}, \qquad (15)$$

where $0 \geq \omega_{u,f,k} \geq 1$ and σ is an optional weight vector to enable different influence of the weight components.

Finally, all friend weights are normalized such that $\sum_{f \in \mathbf{F}_u} \omega_{u,f,k} = 1$ for each u and k, where $\mathbf{F}_u$ is the set of u's friends. The total weight of friend f may then be obtained as $\omega_{u,f} = \sum_k \omega_{u,f,k}$. If a user has a large number of friends, we limit the number of friends that are considered for the collaborative user model by selecting top-n friends based on the total friend weight.

Creating the Collaborative User Model. After obtaining the weight of each friend of u, we construct the *Collaborative* User *Model* (CM). Basically, we take the weighted average of all the individual user models of u's friends. The topic-specific language model for topic k within in the collaborative user model is estimated as follows

$$\theta^{CM}_{u,k,w} = \sum_{f \in \mathbf{F}_u} \omega_{u,f,k}\ \theta^{IM}_{f,k,w}. \qquad (16)$$

The collaborative user model can now be integrated with the individual user model as follows

$$\begin{aligned}\hat{\theta}^{IM.CM}_{u,k,w} =& (1-\lambda)\Big(\beta\theta^{IM}_{u,k,w}\theta^{IM}_{u,k} + (1-\beta)\theta^{CM}_{u,k,w}\theta^{CM}_{u,k}\Big) \\ &+ \lambda P(w|\phi^{TM}_k)\eta,\end{aligned} \qquad (17)$$

where β is a parameter that controls the influence of CM on the IM. We adopt Dirichlet prior smoothing, which allows to smooth sparse individual models more aggressively than rich individual models. In this method, β is defined as

$$\beta = \frac{|M_u|}{|M_u| + \mu}, \qquad (18)$$

where μ is the Dirichlet smoothing parameter. In plain words, we smooth the individual model using the collaborative model and finally smooth using the topic model. Figure 2 illustrates the collaborative user modeling method.

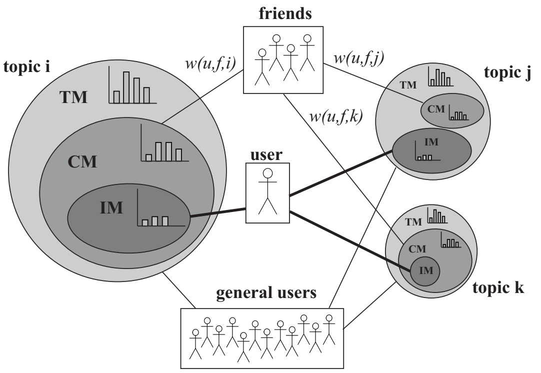


Figure 2: Collaborative User Model

The collaborative personalized ranking function is then defined as

$$P(D,Q,u) \propto \Big(\sum_{k=1}^{K} P(Q|\hat{\theta}^{IM.CM}_{u,k,w})P(D|\hat{\theta}^{IM.CM}_{u,k,w})\Big)P(D). \qquad (19)$$

4.4 Modeling Search Behavior

In addition to analyzing the user's individual microblog content and building the collaborative model, we also model the user's search activity and construct the *Search* User *Model* (SM). As implicit evidence of the user's search interests, we mainly consider search queries issued by the user and the user's feedback on the search results. In microblogs, there are several ways a user can provide implicit relevance feedback. These include re-tweeting (re-sending) or "favoriting" an interesting tweet, or clicking a URL within the tweet. We refer to these actions as *clicks* for convenience. Admittedly, a more thorough analysis of the importance of various click types in user preference modeling is an interesting future work.

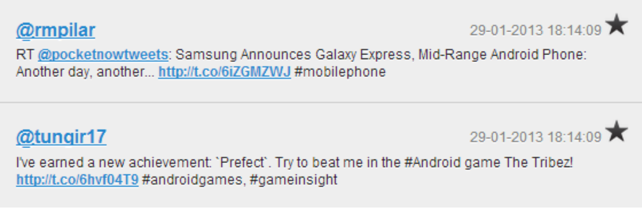
Let $click(u,Q,D)$ denote a click by user u on document D returned to query Q. The set of all clicked documents by u is denoted $\mathbf{S}_u$. For each clicked document D, we first assign D to topic k using Equation (7). Second, we obtain the following implicit relevance feedback from the user's click:

- Topic level feedback $\theta^{SM}_{u,k}$: user's search bias towards topic k. The value of $\theta^{SM}_{u,k}$ is estimated by substituting $\mathbf{S}_u$ in Equation (9).
- Topic-word level feedback $\theta^{SM}_{u,k,w}$: user's preference for words in topic k. This value is estimated by substituting $\mathbf{S}_u$ in Equation (8).
- Query-topic feedback $\theta^{SM}_{u,k,Q}$: user's preference for topic k when issuing query Q. We estimate this value as the maximum likelihood of a click in topic k among all topics clicked for query Q.

The search user model can now be integrated with IM and CM by a weighted sum as follows

$$\begin{aligned}\theta^{IM.CM.SM}_{u,k,w} =& (1-\lambda)\Big[\big(\beta\theta^{IM}_{u,k,w}\theta^{IM}_{u,k} + (1-\beta)\theta^{CM}_{u,k,w}\theta^{CM}_{u,k}\big) \\ &+ \gamma\theta^{SM}_{u,k,w}\theta^{SM}_{u,k}\Big] + \lambda P(w|\phi^{TM}_k)\eta,\end{aligned} \qquad (20)$$

[2] This information can be obtained from, e.g., http://twittercounter.com/pages/100



@rmpilar 29-01-2013 18:14:09 ★

RT @pocketnowtweets: Samsung Announces Galaxy Express, Mid-Range Android Phone: Another day, another... http://t.co/6iZGMZWJ #mobilephone

@tungir17 29-01-2013 18:14:09 ★

I've earned a new achievement: `Prefect`. Try to beat me in the #Android game The Tribez! http://t.co/6hvf04T9 #androidgames, #gameinsight

Figure 3: Evaluation user interface, showing results for query "android".

where γ is a parameter to control the influence of the SM. Parameter setting is discussed in more detail in Section 5.3.2.

Incorporating query-topic feedback. When $\theta^{SM}_{u,k,Q} > 0$ for topic k and a query phrase Q, we may replace $\theta^{SM}_{u,k}$ in Eq. 20 with $\theta^{SM}_{u,k,Q}$ to incorporate the user's query-topic feedback.

The full ranking function for collaborative personalized search is defined as

$$P(D, Q, u) \propto \Big(\sum_{k=1}^{K} P(Q|\theta^{IM.CM.SM}_{u,k,w}) P(D|\theta^{IM.CM.SM}_{u,k,w}) \Big) P(D). \tag{21}$$

By using the ranking function in Equation 21, we incorporate all three user models (IM, CM and SM) in a principled manner. Moreover, our method allows to maintain each user model separately, which has the advantage of fast updatability, and enables the model parameters (i.e., β, γ, λ) to be updated at any time.

Notably, our framework is flexible enough to incorporate additional document-specific and author-specific features in the ranking function, by means of the prior document probability $P(D)$. However, a comprehensive study of document and author features is not within the scope of this paper.

5. EXPERIMENTAL EVALUATION

5.1 Datasets

5.1.1 Background Twitter Corpus

To train the global topic model (TM), we obtained a sample of public tweets from Twitter's Streaming API[3]. We crawled a total of 44.5 million tweets over the course of 6 months in 2013. After filtering non-English language tweets and removing tweets of less than 20 characters in length, our dataset contained 11.7 million tweets. This dataset is used to train the global topic model in Section 5.3.1.

5.1.2 Twitter Search Query Logs

To evaluate the effectiveness of different personalization approaches for Twitter search, a query log with associated information about the user (incl. user's tweets and social connections) is needed. However, such information is not available in commonly used datasets (e.g., the TREC Microblog Track[4]). Therefore, we developed a web-based Twitter search middleware to collect user's search queries and relevance judgements. Users can log in to the system using their Twitter account. Given a search query, the system connects to Twitter's Search API[5] and retrieves 50 recent tweets. The results consist of 3 'popular' tweets as determined by Twitter and up to 47 'general' tweets matching the query. Our system presents all results to the user in a random order, in order to avoid any bias. The user may evaluate the relevance of each result by clicking on a star icon, as shown in Figure 3. The system stores all submitted queries, retrieved tweets and the user's relevance ratings.

Table 1: Query Log Statistics

	Log_CoS	Log_IwS
No. of queries	174	235
Avg. queries per user	15.8	9.42
No. of retrieved tweets	13,712	15,669
No. of relevance judgements	1,251	4,054
Avg. relevant tweets per query	7.19	17.94

Query Log 1: Controlled User Study (Log_CoS). We obtained a search query log with relevance judgements in a user study involving 11 active Twitter users. The user study is divided into two days (referred to as Day 1 and Day 2). On Day 1, users are asked to prepare 10 queries about their topics of interest. The 10 queries are categorized into four types: recency, topical, entity-oriented and ambiguous. The first three types correspond to common search scenarios in microblogs, as reported by Teevan et al. [24]. Additionally, we also consider query ambiguity, which serves as an important motivation in classic personalization research [6]. We note that each query can be classified under multiple types. The query types are detailed as follows:

- *Recency-oriented.* Queries for which relevant tweets must be very fresh (e.g., a search for the results of a football match). In contrast, *non-recency* queries are those for which relevant tweets don't need to be completely new (e.g., "good bar in New York").
- *Topical.* Queries related to the user's long-term interests. For example, an IT professional may issue a query "java" to search for content related to programming.
- *Entity-oriented.* These queries aim to find information about specific named entities, such as people, organizations, products or locations.
- *Ambiguous.* An ambiguous query may have multiple meanings. For example, "java" may refer to a programming language or to an island.

Users are asked to choose at least one query of each type and 10 queries in total. Users are then asked to submit each query in the evaluation system, review the 50 tweets returned by the system and mark relevant tweets.

On Day 2 of the study, users are asked to choose a new set of queries by re-submitting 5 queries from Day 1 and choosing 5 new queries. Similarly to Day 1, users submit each query in the evaluation system and mark relevant tweets.

We present overall statistics of the obtained query log, referred to as *Log_CoS*, in Table 1. We further examine the type of each query, which has been indicated by the users. Table 2 shows the proportion of queries of each type and example queries from the log.

To learn about the topical diversity of queries in the dataset, we manually inspect each query and assign a topical category. We utilize the Yahoo taxonomy[6] and classify queries into the top-level categories. For ambiguous queries, we choose a category based on which tweets were marked as 'relevant' by the user. The distribution of queries by their category is shown in Table 3.

[3] https://dev.twitter.com

[4] https://sites.google.com/site/microblogtrack/

[5] https://dev.twitter.com/docs/api/1.1/get/search/tweets

[6] http://dir.yahoo.com/

Table 2: Overview of query types from Log_CoS and Log_IwS, including the proportion of queries ("%Qrs") and example queries of each type.

Log_CoS

Query type	*%Qrs*	*Example queries*
Recency	44.3%	Boeing 787, iWatch, tv shows, Gun control
Topical	39.1%	Humber bridge, Icing sugar, table tennis
Entity	31%	Obama, NASA, Santa Fe, Lance Armstrong
Ambiguous	33.7%	hostages, galaxy, apple, giants, flash

Log_IwS

Query type	*%Qrs*	*Example queries*
News	9.8%	typhoon hk, Kugan case, air crash, xinjiang riot
Topical	49.2%	hong kong food, windsurfing, stock market
Entity	26.1%	Google, david beckham, Nike, Syria
Ambiguous	25.6%	Chelsea, Surface, langham, simple plan

Table 3: Overview of queries by query category, including the proportions of queries ("%Qrs").

Log_CoS		Log_IwS	
Category	*%Qrs*	*Category*	*%Qrs*
Business & Economy	22.5%	Business & Economy	30.5%
Regional	15.9%	Entertainment	15.8%
News & Media	13.0%	News & Media	9.9%
Computer & Internet	10.1%	Regional	8.9%
Society & Culture	8.0%	Recreation & Sports	8.4%
Entertainment	7.2%	Education	5.9%
Recreation & Sports	7.2%	Society & Culture	4.9%
Education	2.2%	Computer & Internet	3.4%
Science	2.2%	Government	3.0%
Other	11.6%	Other	9.4%

For the purpose of our evaluation, query log data from Day 1 is treated as the *training dataset* and data from Day 2 is treated as the *testing dataset*.

In addition to the query log data, we also crawl the users' Twitter data. Specifically, we obtain the latest 200 tweets of each user and crawl the tweets of the top-20 friends, ranked by friend weight (cf. Section 4.3).

Query Log 2: In-the-Wild User Study (Log_IwS). To obtain users' search preferences in an unrestricted setting, we invite 24 users and conduct an open user study over a 3-month period. Users are invited to use our evaluation system and submit search queries of their choice. As an approximate guide, we ask users to submit at least 10 queries over the evaluation period. When a query is submitted, the user is asked to read through all tweet results and provide relevance ratings.

The statistics of the obtained dataset are given in Table 1. We find that users submitted 9.42 queries on average during the study period, with a standard deviation of 3.15. We also observe that users identified 17.94 relevant results per query, which is higher than 7.19 in the controlled study.

To gain more insight into users' choice of queries and to compare against the Log_CoS dataset, we empirically analyze the query log. First, we perform a post-hoc assignment of queries to the four query types utilized for the Log_CoS dataset. However, we note that the importance of query *recency* may vary among different users, which prevents an objective decision whether a query was recency-oriented or not[7]. Therefore, we instead focus on identifying "news-related" queries, since such queries have a strong recency focus and can be identified more objectively. Table 2 shows the proportion and examples of queries of each type.

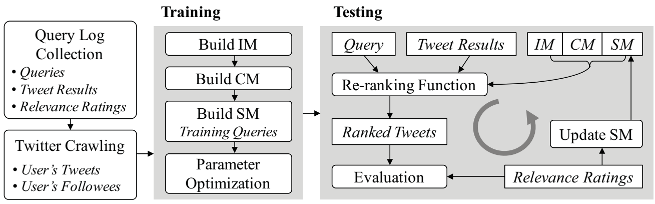


Figure 4: Evaluation Methodology

Similarly to Log_CoS, we analyze the topical diversity of queries and manually classify queries into topical categories. The distribution of queries by category is shown in Table 3.

5.2 Evaluation Methodology

5.2.1 Evaluation Setup

Figure 4 shows an overview of the evaluation process. First, we collect query logs (cf. Section 5.1.2) and crawl Twitter data for each user. Second, we estimate the individual and collaborative user models described in Section 4. Using the training query log, we estimate the search user model (cf. Section 4.4) and tune the global parameters of our framework. Third, we use the testing query logs (i.e., Day 2 of Log_CoS and all queries in Log_IwS) to measure ranking performance. The testing process involves all aspects of our framework, which includes dynamic updating of the search model (SM). For each testing query, we produce a ranking using our framework, evaluate the ranking using relevance judgements from the query log and update the SM. This cycle is repeated for each testing query. The process simulates the behavior of our framework in a real usage scenario, in which a user submits a query, reads through the results and clicks on (e.g., re-tweet) the relevant ones.

5.2.2 Overview of Models and Baselines

We implement the following non-personalized and personalized *baseline models*:

- **Query Likelihood (B-QL).** Standard language modeling approach (cf. Eq. 3), used as a baseline non-personalized model. Used for ranking tweets in [14].
- **Topic Model-based IR (B-TM).** Ranking based on the Topic Model and a background language model. In this method, results are scored using Eq. (5).
- **Personalized Search (B-PS).** Personalized ranking based on a single-layer user language model. This approach is used for personalizing Web search (e.g., [19, 22]). Ranking is determined by the KL-divergence between the personalized query LM $\theta_{q,u}$ and the document LM θ_d.
- **Collaborative Search (B-CS).** This model considers the preferences of a group of users, which is the basis of collaborative Web search [17]. We implement the method in Xue et al. [27], which utilizes a LM of a cluster of users for document ranking.
- **Collaborative Personalized Search (B-CPS).** We implement a model for collaborative personalized Web search by Xue et. al [27]. In this approach, both the user's LM and the collaborative LM are integrated.

[7] For example, user A may be interested in the latest news about "Johnny Depp" (recency query), while user B may want to find both old and new stories about the actor's life (non-recency query).

Table 4: Model parameters

Param.	*Description*	*Eq.*	*Value*
λ	Weight of TM in J-M smoothing	(11)	0.2
μ	Dirichlet prior for CM	(18)	70
γ	Parameter of Search Model	(20)	20
ρ	Parameter for exp. time decay	(12)	0.01

Proposed models. We evaluate each component of the proposed Collaborative Personalized Twitter Search framework:

- **CPTS-IM.** Ranking using the Individual User Model (cf. Equation 11).
- **CPTS-CM.** Ranking using the Collaborative User Model (cf. Equation 17, where $\beta = 0$).
- **CPTS-SM.** Ranking using the Search Model. This method uses SM only in Equation 21.
- **CPTS-All.** Full Collaborative Personalized Search Model (cf. Equation 21).

5.2.3 Metrics

Ranking performance is evaluated using two standard metrics, namely Normalized Discounted Cumulative Gain (NDCG) and Mean Average Precision (MAP).

5.3 Model Training

5.3.1 Topic Model

To train our global topic model, we use the Twitter corpus described in Section 5.1.1. We utilize an online inference algorithm for LDA [9], which is based on stochastic variational inference and allows for processing of massive and streaming data. It was shown in [10] that LDA trained on grouped tweet-documents performs better than training on individual tweet-documents. In our public tweet sample, it is not practical to group tweets by their authors. Instead, we select all tweets containing one or more hashtags and group each tweet to their respective hashtags. In this way, we obtain longer and more semantically-rich "hashtag-documents" for training. As an additional pre-processing step, we remove spam-like hashtag-documents[8] and hashtag-documents of short length.

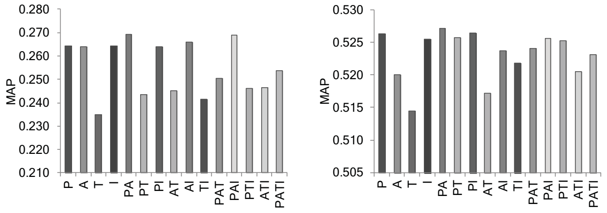
5.3.2 Parameter Setting

We adopt a parameter selection approach commonly used in probabilistic frameworks (e.g., in [8]). Using the training dataset from Section 5.1.2, we optimize the global parameters for our framework. We proceed by optimizing one parameter at a time, while keeping all other parameters fixed. The obtained parameter values are listed in Table 4.

5.4 Analysis of Weight Factors in CM

The collaborative user model constitutes an integral and non-trivial part of our framework. Intuitively, the criteria for selecting which content to include in the CM will largely influence the CM's ranking effectiveness. Therefore, we first study the importance of the friend weighting factors (popularity, affinity, topic-interaction and topic bias) proposed in Section 4.3. We are interested in finding which factor or combination of factors yields the best results.

We build 15 versions of the CM for each user, based on all combinations of the 4 weight factors. We then measure the ranking performance of each CM version in turn, using the CPTS-CM ranking model. Figure 5 shows the results of this experiment on both query logs.



(a) Dataset Log_CoS

(b) Dataset Log_IwS

Figure 5: Performance of the Collaborative Model with different weight factors: Popularity (P), Affinity (A), Topic-Interaction (I), Topic Bias (T).

When using a single weight to build the CM (i.e., 'P', 'A', 'T', 'I' in Fig. 5), the results suggest that popularity and topic-interaction weights are the most effective. This suggests that it is beneficial to assign a higher weight to popular friends, as well as selectively promoting content in topics and by friends with whom a user often engages. The affinity weight is effective on the Log_CoS dataset, but performs poorly on Log_IwS. However, topic bias ('T') shows the weakest performance on both datasets. This suggests that it is not beneficial to assign an 'apriori' weight to all content in a particular topic produced by user's friends. Among all versions of CM, the best performance is achieved with 'PAI' on Log_CoS and 'PA' on Log_IwS. These versions of CM are therefore chosen when reporting overall ranking performance in the following sections.

5.5 Results and Discussion

In this section, we evaluate the ranking effectiveness of individual components of our framework and compare them with the baseline methods listed in Section 5.2.2. Additionally, we perform a paired student's t-test to determine if the differences between the results of our methods and each baseline method are statistically significant (p-value<0.05). Table 5 shows the overall ranking accuracy on both query logs, with indications of statistically significant differences over baseline methods.

From the results, we observe that standard language modeling (B-QL) is outperformed by each component of our framework. This result is somewhat expected, given that B-QL does not consider individual users or their social neighborhood. The topic model-based retrieval model (B-TM) shows a superior performance to B-QL, in particular at higher ranks (e.g., NDCG@5). This suggests that incorporating the latent semantics for scoring tweets provides an advantage over standard query likelihood. The personalized and collaborative baseline methods (B-PS, B-CS, B-CPS) fail to outperform the non-personalized baselines on the Log_CoS dataset. On the Log_IwS dataset, they achieve a marginal improvement at lower ranks (NDCG@10 and beyond). This shows that simply applying personalization techniques used in Web search may perform poorly in microblog search. In particular, we note that the collaborative and personalized baseline (B-CPS) does not achieve a cumulative improvement over its individual components (B-PS, B-CS). This further indicates that simply fusing user's individual preferences with the group's preferences may harm ranking effectiveness in the microblog domain.

[8] Example of a spam-like hashtag is "#followback", which is included in automatically generated tweets sent around the social network.

Table 5: Overall ranking results. Statistical significance of each result against the baselines (p-value< 0.05) is denoted using symbols representing each baseline and listed next to the respective baseline's name.

	Dataset Log_CoS					Dataset Log_IwS				
	NDCG@k					*NDCG@k*				
Model	*k=5*	*k=10*	*k=20*	*k=50*	*MAP*	*k=5*	*k=10*	*k=20*	*k=50*	*MAP*
B-QL $^{(\star)}$	0.191	0.214	0.236	0.268	0.244	0.460	0.471	0.486	0.520	0.503
B-TM $^{(\circ)}$	0.236	0.249	0.275	0.304	0.244	0.470	0.479	0.496	0.527	0.507
B-PS $^{(\diamond)}$	0.178	0.202	0.230	0.262	0.229	0.465	0.481	0.496	0.529	0.509
B-CS $^{(\triangleright)}$	0.203	0.221	0.253	0.284	0.239	0.466	0.480	0.496	0.529	0.510
B-CPS $^{(\triangle)}$	0.182	0.204	0.232	0.264	0.230	0.464	0.479	0.495	0.528	0.510
CPTS-IM	0.257 $^{\diamond}_{\triangle}$	0.268	0.289	0.321	0.253	0.472	0.485	0.504	0.534	0.517 $^{\star}_{\circ}$
CPTS-CM	0.250 $^{\diamond}_{\triangle}$	0.267 $^{\diamond}_{\triangle}$	0.292 $^{\diamond}_{\triangle}$	0.319 $^{\diamond}_{\triangle}$	0.269 $^{\diamond}_{\triangle}$	0.485	0.496 $\star$	0.513 $\star$	0.544 $\star$	0.527 $^{\star}_{\circ}$
CPTS-SM	0.251 $^{\diamond}_{\triangle}$	0.272 $^{\diamond}_{\triangle}$	0.295 $^{\diamond}_{\triangle}$	0.323 $^{\diamond}_{\triangle}$	0.279 $^{\diamond}_{\triangle}$	0.508 $^{\star\circ\diamond}_{\triangleright\triangle}$	0.518 $^{\star\circ}_{\triangleright\triangle}$	0.532 $^{\star\circ\diamond}_{\triangleright\triangle}$	0.560 $^{\star}_{\circ}$	0.540 $^{\star\circ\diamond}_{\triangleright\triangle}$
CPTS-All	0.292 $^{\star\circ\diamond}_{\triangleright\triangle}$	0.304 $^{\star\circ\diamond}_{\triangleright\triangle}$	0.321 $^{\star\circ\diamond}_{\triangleright\triangle}$	0.35 $^{\star\circ\diamond}_{\triangleright\triangle}$	0.297 $^{\star\circ\diamond}_{\triangleright\triangle}$	0.513 $^{\star\circ\diamond}_{\triangleright\triangle}$	0.522 $^{\star\circ\diamond}_{\triangleright\triangle}$	0.536 $^{\star\circ\diamond}_{\triangleright\triangle}$	0.564 $^{\star\circ\diamond}_{\triangleright\triangle}$	0.542 $^{\star\circ\diamond}_{\triangleright\triangle}$

Among the proposed models presented in this paper, the IM alone outperforms all baseline models. We note that IM significantly outperforms its baseline counterpart (B-PS) by NDCG@5 on Log_CoS. This demonstrates the effectiveness of our two-level user model structure, which utilizes latent topics in microblogs to organize user preferences.

A further improvement of MAP is achieved by the collaborative user model (CM). However, we observe that CM is not as effective as IM at top ranks (e.g., NDCG@5) on Log_CoS. This indicates that IM may be more effective in promoting relevant tweets to the top positions, if the IM contains sufficient information from the user's tweets. Regarding the results of CM, we note that user's own tweets are not considered within this model. This situation may arise in practice if a user produces little own content, but follows others. This may particularly benefit newly registered users.

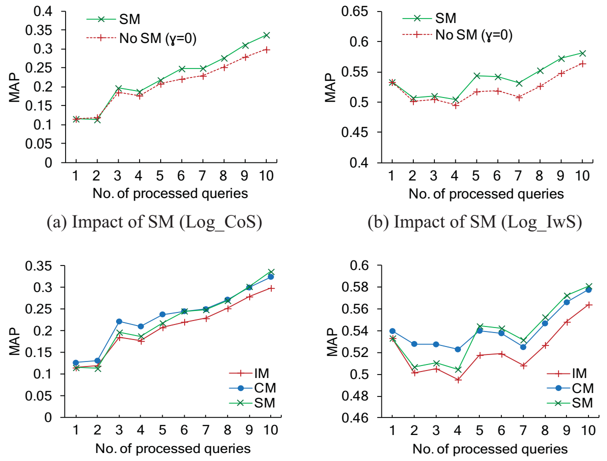
The search user model (SM) shows the best performance among the proposed models. However, we note that the SM is based on the user's implicit feedback and hence faces the cold-start problem. Our framework is designed to overcome the cold-start problem by considering the user's content and social connections in the IM and CM, respectively. The overall results show the strength of the IM and CM, even when no information about the user's search behavior is available.

Finally, the full proposed framework (CPTS-All) achieves the best overall ranking performance. On the Log_CoS dataset, the difference with all baselines except B-TM is statistically significant. On the Log_IwS dataset, we confirm statistical significance compared with all baselines. The results demonstrate that all three information sources in our framework are complementary in improving ranking performance.

5.5.1 Comparison Among Proposed Models

In this section, we further compare the performance of each model in our framework. Since each model uses a different source of evidence about the user's preferences, each model may be effective under different circumstances.

First, we focus on the Search Model (SM). On the one hand, this model is most prone to the cold-start problem when a user first uses the system. On the other hand, the model can be dynamically updated each time a user submits a query and provides relevance feedback (referred to as a *query-feedback step*). We therefore study how the effectiveness of SM evolves with each query-feedback step. For each user, after the i-th query is processed and relevance feedback is received, we calculate the average MAP since the first until the i-th query. In Figures 6 (a) and (b), we show the average per-user MAP after each query-feedback step. We observe that the effectiveness of SM increases with more queries and relevance feedback from the user.



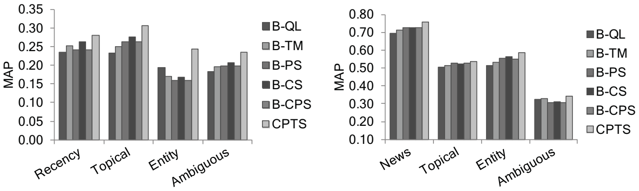
(a) Impact of SM (Log_CoS) (b) Impact of SM (Log_IwS)

(c) Comparison of models (Log_CoS) (d) Comparison of models (Log_IwS)

Figure 6: Average per-user ranking performance after processing i user's queries.

In the next step, we compare the performance among the three proposed models. Similarly to the previous experiment, we calculate the average per-user MAP after each query-feedback step and show the results in Figures 6 (c) and (d). For both query logs, the results suggest that for the first few queries (first 5 queries for Log_CoS and first 4 queries for Log_IwS), CM gives the best results among all models. However, with more relevance feedback, the performance of SM improves, enabling it to outperform CM.

It is important to note that the previous results are averaged over a number of users and queries, which blurs some details about each model's performance. In particular, when inspecting the ranking effectiveness for a query Q, we may determine which of the proposed models achieves the best performance for Q. We therefore measure the "success rate" of each model, in terms of the number of queries for which the model achieved the highest MAP. On Log_CoS, IM, CM and SM achieved the highest MAP for 23.5%, 49% and 27.5% of queries, respectively. On Log_IwS, IM, CM and SM achieved the highest MAP for 19.1%, 31.8% and 49.1% of queries, respectively. From the results, we see that no single model produces the best performance for all queries. This again confirms that all three models are complementary



(a) Dataset Log_CoS

(b) Dataset Log_IwS

Figure 7: Effectiveness for different query types.

and contribute to the overall ranking score when integrated in our framework.

5.5.2 Ranking Performance by Query Types

In this section, we study the effectiveness of our framework when dealing with different types of queries. Intuitively, different types of queries may require personalization to a different extent. In the Web search scenario, it is reported that personalization may even harm ranking quality for some query types [6]. We focus on the four query types described in Section 5.1.2. The proportion of each query type in our datasets is given in Table 2.

Figure 7 shows the average MAP for each query type. Among the personalized and collaborative baselines, we observe that B-CS achieves the best performance for all query types on Log_CoS. However for entity-oriented and ambiguous queries, we find that the performance of B-CS is not very stable across our datasets and fails to outperform non-personalized baselines in some cases. Moreover on Log_IwS, we do not observe significant differences between the personalized and collaborative baselines. These results indicate that the existing methods, which originate from Web search, do not produce satisfactory results for microblog queries.

In contrast, the proposed framework improves the ranking performance for all query types on both datasets. Our method is effective even in cases when the personalized baselines perform poorly. For entity-oriented queries in Log_CoS, we improve the baseline MAP of 0.194 (B-QL) to 0.245, while B-CS only achieves 0.169. For ambiguous queries in Log_IwS, the baseline MAP of 0.327 (B-TM) is improved by our method to 0.341, while B-CS only achieves 0.31.

6. CONCLUSION

In this paper, we present a novel probabilistic framework for Collaborative Personalized Twitter Search. The framework integrates a variety of information about the user's posting and searching preferences. At the core, we develop a topic-sensitive collaborative user model, which utilizes users' social connections to augment the user profile and improve ranking performance. Furthermore, we propose a new two-layer user model structure, which effectively handles the diversity of microblog users' preferences. Our experimental evaluation has demonstrated superior performance against competitive baselines in a variety of settings.

As relevant issues for future work, we plan to categorize the query types that arise in microblogs and design query-dependent personalization strategies. In another direction, we plan to incorporate more features into our framework, such as spatial and temporal dimensions of user preferences.

7. REFERENCES

[1] F. Abel, Q. Gao, G. Houben, and K. Tao. Analyzing Temporal Dynamics in Twitter Profiles for Personalized Recommendations in the Social Web. In *WebSci'11*, 2011.

[2] Y. Cha, B. Bi, C.-C. Hsieh, and J. Cho. Incorporating popularity in topic models for social network analysis. In *SIGIR'13*, 2013.

[3] J. Chen, R. Nairn, L. Nelson, M. Bernstein, and E. H. Chi. Short and Tweet: Experiments on Recommending Content from Information Streams. In *CHI'10*, 2010.

[4] K. Chen, T. Chen, G. Zheng, O. Jin, E. Yao, and Y. Yu. Collaborative Personalized Tweet Recommendation. In *SIGIR'12*, 2012.

[5] W. B. Croft, S. Cronen-Townsend, and V. Lavrenko. Relevance Feedback and Personalization: A Language Modeling Perspective. In *DELOS Workshop*, 2001.

[6] Z. Dou, R. Song, and J.-R. Wen. A Large-scale Evaluation and Analysis of Personalized Search Strategies. In *WWW'09*, 2007.

[7] Y. Duan, L. Jiang, T. Qin, M. Zhou, and H.-Y. Shum. An Empirical Study on Learning to Rank of Tweets. In *COLING'10*, 2010.

[8] M. Efron and G. Golovchinsky. Estimation Methods for Ranking Recent Information. In *SIGIR'11*, 2011.

[9] M. D. Hoffman, D. M. Blei, C. Wang, and J. Paisley. Stochastic Variational Inference. *Journal of Machine Learning Research*, 2013.

[10] L. Hong and B. D. Davison. Empirical Study of Topic Modeling in Twitter. In *SOMA Workshop*, 2010.

[11] A. Java, X. Song, T. Finin, and B. Tseng. Why We Twitter: Understanding Microblogging Usage and Communities. In *WebKDD/SNA-KDD Workshop*, 2007.

[12] K. W.-T. Leung, D. L. Lee, and W.-C. Lee. Personalized Web Search with Location Preferences. In *ICDE'10*, 2010.

[13] X. Li and W. B. Croft. Time-based Language Models. In *CIKM'03*, 2003.

[14] K. Massoudi, M. Tsagkias, M. de Rijke, and W. Weerkamp. Incorporating query expansion and quality indicators in searching microblog posts. In *ECIR'11*, 2011.

[15] T. Miyanishi, S. Kazuhiro, and U. Kuniaki. Improving pseudo-relevance feedback via tweet selection. In *CIKM'13*, 2013.

[16] N. Naveed, T. Gottron, J. Kunegis, and A. Alhadi. Searching microblogs: coping with sparsity and document quality. In *CIKM'11*, 2011.

[17] B. Smyth. A Community-Based Approach to Personalizing Web Search. *Computer*, 40(8):42–50, Aug. 2007.

[18] W. Song, Y. Zhang, T. Liu, and S. Li. Bridging Topic Modeling and Personalized Search. In *COLING'10*, 2010.

[19] D. Sontag, K. Collins-Thompson, P. N. Bennett, R. W. White, S. Dumais, and B. Billerbeck. Probabilistic Models for Personalizing Web Search. In *WSDM'12*, 2012.

[20] B. Sriram, D. Fuhry, E. Demir, H. Ferhatosmanoglu, and M. Demirbas. Short Text Classification in Twitter to Improve Information Filtering. In *SIGIR'10*, 2010.

[21] J.-T. Sun, Z. Chen, H. Liu, and Y. Lu. CubeSVD: A Novel Approach to Personalized Web Search. In *WWW'05*, 2005.

[22] J. Teevan, S. T. Dumais, and E. Horvitz. Personalizing search via automated analysis of interests and activities. In *SIGIR'05*, 2005.

[23] J. Teevan, M. R. Morris, and S. Bush. Discovering and Using Groups to Improve Personalized Search. In *WSDM'09*, 2009.

[24] J. Teevan, D. Ramage, and M. R. Morris. #TwitterSearch: A Comparison of Microblog Search and Web Search. In *WSDM'11*, 2011.

[25] H. Wang, X. He, M.-W. Chang, Y. Song, R. W. White, and W. Chu. Personalized ranking model adaptation for web search. In *SIGIR'13*, 2013.

[26] X. Wei and W. B. Croft. LDA-based document models for ad-hoc retrieval. In *SIGIR'06*, 2006.

[27] G.-R. Xue, J. Han, Y. Yu, and Q. Yang. User language model for collaborative personalized search. *ACM Transactions on Information Systems*, 27:1–28, Feb. 2009.

[28] C. Zhai. Statistical Language Models for Information Retrieval A Critical Review. *Foundations and Trends in Information*