Tweets You Like: Personalized Tweets Recommendation based on Dynamic Users Interests

Shaymaa Khater[1], Hicham G. Elmongui[2], Denis Gračanin[1]
[1] Department of Computer Science,Virginia Tech,USA
[2] Department of Computer and Systems Engineering, Alexandria University, Egypt
skhater@vt.edu, elmongui@alexu.edu.eg, gracanin@vt.edu

Abstract

Microblogs like Twitter have become a significant mean for users to express their opinions or share pieces of interesting news by posting relatively short messages (corpus) compared to the regular blogs. The increasingly large volume of corpus updates that users receive daily is overwhelming. In this paper, we propose a dynamic personalized recommendation system that provides the user with the most important tweets. The proposed system captures the user's interests, which change over the time, and shows the messages that correspond to such dynamic interests. Our performance evaluation demonstrate the effectiveness of the proposed recommendation system and shows that it improves the precision and recall of identifying important tweets by up to 36% and 80%, respectively.

1 Introduction

With the rise of the Internet and blog usage, social media is considered an essential source of providing information by publishing and sharing user-generated content. Twitteris one of the most popular microblogging social media platforms [3]. It was launched in July 2006 and has 284 million monthly active users and 500 million postings per day [1]. Twitter poses a question to its users, "what is happening?" and the answer to this question is restricted to 140 characters.

Twitter users can either post tweets, or can subscribe to updates from other users by following them. They can interact with the stream of tweets they are receiving in their timeline by replying (commenting on a tweet posted earlier by themselves or others), retweeting (resending an earlier tweet posted by other to followers, giving credit to the original publisher) or favoring (liking) the tweets.

Although tweets may contain valuable information, many are not interesting to the users. A large number of tweets can overwhelm users since they interact with many other users and they have to read ever increasing content volume on their timeline [22]. Thus, the difficulty in finding the "matching" users and recommending content that are of interest to users became a key challenge for social networks sites.

Recommendation systems have been proposed to help users cope with information overload by predicting the items that a user may be interested in. We propose a method to identify tweets that may be of potential interest to the user. Since the user's interests in different topics change over time, we focus on studying this change, and recommending to each user the most interesting tweets on the user's timeline at specific time. We study the user's activities and relationships on Twitter and answer research questions about the individual user's interests:

- How can we infer personal interests from the user's Twitter activities and interactions and to what extent do personal interests change over time?
- What are the other features that can be extracted from the user's Twitter activities and can affect the recommendation made to the user?

The problem can be formalized as follows:

Given: A timeline of user X with all tweets posted by his followees, and a history of his own tweets, replies, retweets and favorites.

Goal: Determine if a tweet t is of interest to the user X at the time it is posted.

Some of the related work has been targeting the study of the retweet behavior of the users and the factors that affect predicting the popular tweets in the social network as a whole. Our work is different as it focuses on the factors at the personalized level rather than general ones, and the effect of the change of these factors over time, i.e., building a model that is personalized for each user based on the temporally dynamic features; that is, features that are determined based on the time a tweet is posted.

In this paper, we define a model to classify a given tweet into important or not important. The model mainly considers users' level of interest in the tweets topic at the same time that given tweet is posted. Other features include the authority of the publisher (the number of users following the publisher), the tweet content based features such as the length of the tweet and the retweet count, and the social relation feature which represent the relation between the user and the publisher of the tweet.

The contributions are summarized as follows: *1)* We explore the application of the LDA model in modeling the tweets' topics, and propose an extension to extract better topics in microblogging environments. This relies on aggregating tweets and presenting them in a a bigger document size to enhance the topic modeling from microblogs; *2)* We

 ISBN: 978-1-62561-003-4

introduce the notion of dynamic Level of Interest (LoI), in which we build a user specific time variant (dynamic) level of interest graphs for each topic of the tweets' topics. This is based on utilizing the weights of topics in the user's tweets to determine its level of importance to the user; *3)* We introduce a model that incorporate the dynamic change in users' interests in topics, along with other social features, tweets related features to recommend interesting tweets for the user; *4)* We conduct extensive experiments to analyze the behavior of the LDA topic model and identify some factors that can affect its performances such as, the length of the document and the number of topics used in the topic modeling process.

The paper is organized as follows: Related work is discussed in Section 2. We introduce a general overview for our approach in Section 3. The complete description of the model and its components is found in Section 4. The experimental results are discussed in Section 5. Conclusions and future work are described in Section 6.

2 Related Work

As our approach focus on tweet recommendation and topic modeling, we provide an overview of the related research.

2.1 Recommendation Systems in Twitter

To overcome the information overload problem in online social networks, many recommender systems were introduced to help users find interesting information. Some of these systems were conducted to study the social network structure and recommend friends to the user based on the similarities of interests. One study proposed Twittomender that recommends Twitter users based on the relationships of their Twitter social graphs [11]. Kwak estimated the influence of users on Twitter by proposing three methods: the number of followers, PageRank and the number of retweets [12]. Other systems studied the personalized recommendation systems to recommend only useful content to users. Chen et al. proposed a collaborative filtering method to generate personalized recommendations in Twitter through a collaborative ranking procedure [9]. Pennacchiotti et al. proposed a method to recommend tweets to users by following users' interests and using the tweet content [19]. Chen et al. proposed a URL recommendation model to demonstrate the utility of various combinations of tweet content and social graph information during recommendation [8]. These methods are different from our work as they miss the dynamic change of interests of the user.

Limited work has been done in dynamically personalized tweet recommendation. The study done by Abel et al. is most relevant to our problem [5]. Abel et al. analyzed how users profiles changes over time, and how to recommend news articles for topic based profiles. Our model is different in that it tries to capture the change of each user's interest in different topics over time, and recommend interesting tweets based on this interest. Uysal et al. [24] explored user-publisher and user-tweet features to rank the twitter feed for each user based on their probability of being retweeted. Compared to our model, it just uses the explicit features of the tweets without considering the personalized features for each user.

2.2 Topic Modeling

Topic Modeling is a rapidly-growing field of research in the area of text mining, and statistical modeling. As text comprises about 85% of data worldwide [6], topic models have been widely used to address the problem of 'information overload' associated with this huge collection of text and corpuses. They are also extended in many ways to be used in social media to identify text patterns in the content and to facilitate many applications such as sentiment analysis and content filtering [15]. As our model uses LDA for topic modeling, it will be briefly described in Section 4. Although topic models such as Latent Dirichlet Allocation (LDA) had been applied successfully on different articles and documents, their application on microblog contents such as Twitter faces different challenges. *1)* The posts are short, 140 characters; *2)* The use of informal language and nonstandard abbreviations (e.g. LOL, WOW); and *3)* The text contain other context that may be act as a noise as the URL, Twitter names and tags. To overcome these difficulties, some studies proposed to aggregate all the tweets of a user in a single document [26]. This can be regarded as an application of the author-topic model [23] to tweets, where each document (tweet) has a single author. However, this treatment assumes that the user's interests in topics will not change over time, which contradicts our assumption. Therefore, we are proposing a complimentary approach to the LDA model that can help in extracting better topics from microblogs.

3 Problem Description

When users login on Twitter, they see a stream of tweets sent by friends which composes their timeline. Many of these tweets are conversational tweets and/or are not of personal interest to the user. The goal of our model is to decide for each user the tweets that might be of interest from the user's timeline. Beside being able to post their own tweets, users can also interact with their timeline by replying, retweeting or favoring the tweets. As there are no explicit means to extract the user's level of interest in a tweet from Twitter, we relied on these actions to predict the user's interests. Hence, the retweets, replies and favorites can be used as an indication of the interest of the user in the corresponding tweets. We define a tweet as a tuple $\langle \mathbf{u}, \mathbf{p}, \mathbf{e}, \mathbf{o}^e, t, int_{tu} \rangle$ where (vectors are indicated by boldface :

- $\mathbf{u}$ is a vector describing the features of the user u receiving the tweet.
- $\mathbf{p}$ is a vector describing the features of the publisher p of the tweet.
- $\mathbf{e}$ is a vector describing the features of the tweet e.
- $\mathbf{o}^e$ is a vector that holds the distribution of probabilities of the tweet text e across different topics.

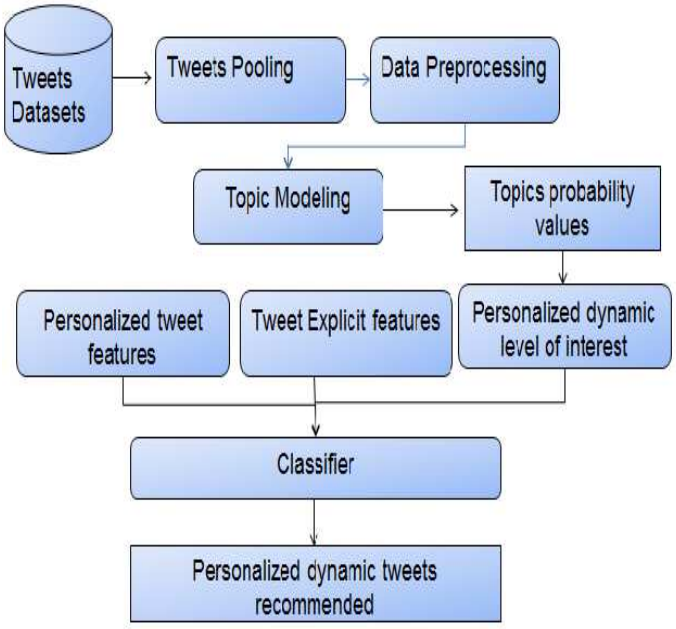

Figure 1: System framework.

- t is the time window in which the tweet is posted.
- int_{tu} is a binary value indicating whether this tweet is of interest to the user u or not.

Given these tuples for the tweets in the user history, our goal is to predict int_{tu} for each new tweet.

4 Personalized Recommendation System Model

We now describe in more details our approach and discuss its main components, as shown in Figure 1.

4.1 Tweets pooling

The short length of tweets might result in a poor topic model. Thus, to help get around the problems associated with the analysis of numerous small documents, we construct large documents out of the tweets. So, instead of looking at each tweet individually, we group together tweets that are similar in some sense (same semantics, same hashtags, etc.) in a process called *pooling*. In our model, we present some schemes that we used to aggregate tweets into a larger documents from which a better topic model can be trained. These pooling schemes can be described as follows:

1. **Hashtag pooling**: In Twitter, the Hashtag is a string of characters preceded by the hash # character. They are used as identifiers for tweets discussing the same topic [13]. By including hashtags in a message, users indicate to which conversations and topic their message is related to. Using these hashtags can be a good indicator for tweets relatedness, and so can be used in the aggregation process of tweets. For the hashtag-based pooling scheme, we aggregated documents sharing each hashtag in one pool, as in [17]. If a tweet has more than one hashtag, this tweet will be added to the tweet-pool of each of those hashtags.

2. **Replies pooling**: We used replies for tweets as another way for aggregation. In general, a reply is a string preceded with the @ character. It is used as a comment on another tweet posted by you or by anybody in the social networks. As the tweets and their replies might be sharing the same topics discussed, so aggregating them in one pool can be another good indication for the tweet relatedness.

3. **URLs pooling**: We also aggregated tweets that include the same URLs in their text. Tweets sharing the same URLs might be discussing the same topic, and hence can be aggregated.

4.2 Topic Extraction of Tweets

In order to predict the user's interest in a corpus, we based our prediction on the user's interest in the topic(s) covered in that corpus, alongside with other features. Consequently, we needed to build a Topic Model of our tweets. Topic models, such as latent Dirichlet allocation (LDA) [7], are well-known for exploratory and predictive analysis of text. Generally topic models define topics as distributions over the words in a vocabulary and documents as being generated by mixtures of these topics. Topic models represent document words in a bag-of-words format without considering word order to be of any particular importance. According to the frequencies of different words appearing together in each document, the model then determines the most relevant set of words to each topic. After training a topic model, it can be used later to infer the topic(s) available in new documents.

Formally, the Latent Dirichlet Allocation (LDA) Model can be described as follow:

Given: A set of e posts denoted by $\mathbf{E} = \{e_1, \ldots, e_n\}$, the LDA algorithm generates a set of k topics denoted by $\mathbf{L}$= $\{l_1, ..., l_k\}$. Each topic is a probability distribution over m words denoted by $l_i = \{w_1^i, \ldots, w_m^i\}$ where w_j^i is a probability value of word j assigned to topic l_i. The post can then be represent as $\mathbf{o}^e = \{o_1^e, ..., o_k^e\}$ where o_j^e is the percentage of topic l_j in the post e composition.

In the case of our model, each aggregated pool of tweets is considered a document, and the words in the pool are considered the vocabulary. We use these documents and the vocabulary to extract the topics that form the corpus.

4.3 Dynamic Level of Interest

In this section, we study how the interests of individual users about a certain topic change over time. Getting the dynamic level of interest in a tweet takes place through some steps:

1. First we get the per topic activity in each day d for the user, denoted by $\mathbf{A}_d = \{a_1^d, \ldots, a_k^d\}$ where a_i^d is the level of activity of the user in topic l_i on day d. $\mathbf{A}_d$

Algorithm 1 Calculate Users Level of Activity Per Topic

```
procedure CalculateDailyActivityVectors
input Users: Set of all users
begin
  L ← List of all topics
  for each User u in Users do
    Days ← All Days in which u was active
    for each Day d in Days do
      Tweets ← All tweets by u in d
      // Initialize vector for user u in day d
      A_d^u ← [0, ..., 0]
      for each Tweet e in Tweets do
        o^e ← Percentages of topics in e
        for each Topic l in L do
          A_d^u[l] = A_d^u[l] + o^e[l]
        end for
      end for
    end for
  end for
end
```

is calculated by adding the vectors $\mathbf{o}^e$ in that day, as in Equation 1. The details of this step are shown in Algorithm 1.

$$\mathbf{A}_d[i] = a_i^d = \sum_{\forall e \in E : e_{date} = d} \mathbf{o}^e[i] = \sum_{\forall e \in E : e_{date} = d} o_i^e \quad (1)$$

2. Given a new tweet e_{new}, the user's level of interest in the tweet can be calculated using Equation 2. Basically, the equation sums up the user activity vectors in the window of last seven activity instances prior to the tweet creation day d. Each of these instances corresponds to user's actions done in one day. For a user who is active (posting a tweet, replying, retweeting or favoring another tweet) every day, this window will span one week period. For less active users (not active every day), this window will be longer to cover the last seven active days in which the user was active. We only consider the last seven instances, as considering intervals longer than seven days will introduce irrelevant noisy tweets as discussed in [21]. his step is illustrated in Algorithm 2.

$$\begin{aligned} LevelofInterest(u, e_{new}) &= \sum_{l \in L} (\mathbf{o}^{e_{new}}[l] \cdot \sum_{d \to d-7} \mathbf{A}_d[l]) \\ &= \sum_{l \in L} (o_l^{e_{new}} \cdot \sum_{d \to d-7} a_l^d) \end{aligned} \quad (2)$$

4.4 Personalized Tweet Recommender

In addition to measuring the dynamic level of interest for each user, some other static features can affect his interests. Some of these features represent the personalized interests of the user, others are general features that are related to the tweet‘s quality or the publisher‘s authority that can affect the tweet‘s degree of interest to the user. The following sections describe the personalized features and other explicit features that might affect user's interest.

Algorithm 2 Calculate Level of Interest in a new Tweet

```
function CalculateLevelOfInterest(User u, Tweet e)
begin
  o^e ← Percentages of topics in e from LDA model
  d ← e posting date
  LoI ← 0
  for each Topic l in L do
    val ← 0
    for i = 1 to 7 do
      val ← val + A_{d-i}^u[l]
    end for
    val ← val * o^e[l]
    LoI ← LoI + val
  end for
  return LoI
end
```

4.4.1 Personalized Social Features

Social features are the features that represent the social relationship between the user and the publisher. This relation can be friendship, neighborhood who posts tweets about events happening in the neighborhood or celebrities who have interests in common with the user. These social features include

- User-publisher similarity: this feature measures the similarity between activity level of the user and the publisher on all topics. This is measured as the cosine similarity between vectors formed by summation of the level of interest in a topic for the user over time. This is shown in Equation 3. Generally, the cosine similarity measure yields a value between -1 and 1. The value of 1 means the exact distribution match, i.e., activities of both users are distributed in the same proportions on different topics, though one of them might be generally more active than the other. The value of 0 means that users have nothing in common.

$$\begin{aligned} CosineSimilarity(U_t, P_t) &= \frac{U_t \cdot P_t}{\|U_t\| \cdot \|P_t\|} \\ &= \frac{\sum_{t=1}^{T} U_t \times P_t}{\sqrt{\sum_{t=1}^{T} {U_t}^2} \times \sqrt{\sum_{t=1}^{T} {P_t}^2}} \end{aligned} \quad (3)$$

4.4.2 Explicit Features

Besides the personalized social features, we analyzed other explicit features that can affect the user‘s interests. These features appear or can be inferred from the user profile. This includes:

- Publisher based features: related to the tweets's publisher. These features are used as an indicator for the activity of the publisher:

  - Publisher followers: the number of followers for each publisher. High authoritative publishers are likely to have more followers than others.
  - Publisher tweets count: the number of tweets posted by the publisher since the opening of the account. This feature is an indicator for how active the publisher is.
  - Mention count: the number of times a publisher's name is mentioned in all the tweets. If a publisher is frequently mentioned, the publisher is more likely to be popular and has more interactions than other publishers.

- Tweet based features: describe the tweets contents as:
  - Retweet count: the number of times the tweet got retweeted. It is a way of estimating the popularity of the tweet. A tweet retweeted more times is more likely to be a useful one.
  - HasURL, HasHashtag: sometimes a publisher includes supplement to their tweets with URL or hashtags. Hashtags can sometimes be an indication of the tweet's topic.

- Location feature: represents the cities or countries found in the publishers profiles. This feature is used to capture the spatial neighborhood effects. If a publisher posted a tweet about local events, and this publisher is the user's neighbor, then most probably the user will be interested in this post.

5 Experiments

In this section, we describe our datasets and the preprocessing steps followed by the experimental results for each step in our model.

5.1 Dataset

For our experiments, we created a Twitter data set containing five million tweets and 20 thousands users that were seeded by first selecting 100 active users from the Virtual Town Square blog [4]. We used Twitter REST API [2] to facilitate the data collection. The majority of the tweets collected were published in a three-months period from April 2013 to June 2013. We then expanded the user base by following their followers and friends. We were able to include 20 thousands users with all their posts.

As Twitter APIs does not allow access to the timeline of the user directly without authorization, we build each user's timeline by first getting the posts for each of the base users, and then following the tweets posted by their friends, and consider them the scanned tweets by the user. All the favored tweets by the base users are also retrieved.

5.2 Preprocessing

We build our model from a repository of more than five million tweets. To eliminate incomplete and noisy data, we preprocessed the tweets by discarding tweets with non-English words. We also removed meaningless words such as stop-words, Twitter specific stop words, user names, and special characters and stemmed the remaining words.

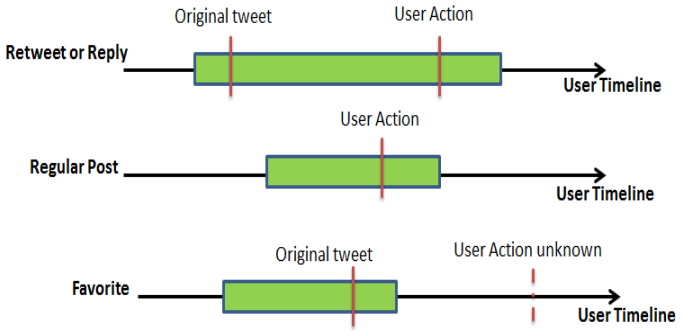

Figure 2: Timeline window for the user.

Usually users do not have time to see all the tweets posted on their timeline. Also, users can be away or inactive (i.e., no posts, retweets or favorites) for long periods of time. Considering this period in our dataset will cause the number of negative examples to be much bigger compared to the number of positive examples.

To overcome this, we filtered the browsing history by considering a window made up of a set of 20 tweets. The number 20 is chosen due to the fact that Twitter limits the number of tweets to be retrieved to 20 tweets each time the user browse his timeline. The window's sliding scheme depends mainly on the user's action in time of browsing the history tweets. As in the case of retweeting or reply, the window of interest will be 15 tweets before the original tweet till five tweets after the user's action. When the user is just posting a tweet without referencing any history ones, the window of interest will be considered 15 tweets before and five tweets after the user's action, respectively. Twitter API does not reveal the exact date of favoring a tweet. In the case of favoring a certain tweet, the window of interest is chosen to be 15 tweets before and five tweets after the original favored tweet.

Figure 2 shows the different cases for choosing the window of interest in the user's timeline. This filtration step is applied to the tweets before the classification step in both training and testing. The filtered out tweets are still used in training the topic model and calculating the user activities.

5.3 Tweets Pooling

The tweets pooling process aggregates tweets that are semantically similar into one pool. Each pool is then treated as a document. For our dataset, we first began by aggregating each tweet and their replies into one pool. Then, for each hashtag, we aggregated all the tweets that are sharing the same hashtag. Finally, we aggregated the tweets that contain the same URLs in their content.

This pooling process decreases the number of documents and increases the document size to be the size of the aggregated tweets. Table 1 shows the number of pools generated from each pooling schema along with the number of tweets in the largest pool in each scheme.

Table 1: Characteristics of different pooling scheme

Pooling scheme	Number of pools	Largest pool size
Unpooled	5741434	1 Tweet
Replies pooling	5660386	51 Tweets
Hashtags pooling	4688744	21483 Tweets
URLs pooling	4546896	3364 Tweets

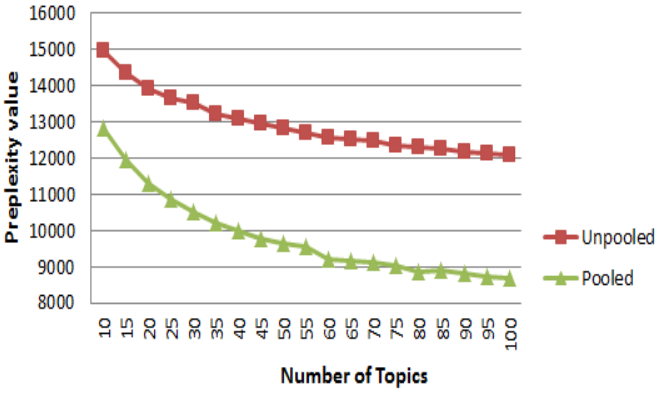

Figure 3: Perplexity for LDA and our model.

5.4 Evaluating Topic Models

The unsupervised nature of topic modeling methods makes choosing one topic model over another a challenging task. Topic model quality tends to be evaluated by performance in a specific application. Topic models can be evaluated based on perplexity [25] as a quantitative method. Perplexity is a well-known standard metric used in Information Retrieval field. It tries to quantify the accuracy of a model by measuring how well the trained model deals with an unobserved test data as in Equation 4. Perplexity is defined as reciprocal geometric mean of per word likelihood of a test corpus. A lower perplexity indicates a better generalization performance.

$$Perplexity(D_{test}|M) = \exp \frac{-\sum_{d \in D_{test}} \log P(w_d|M)}{\sum_{d \in D_{test}} N_d} \quad (4)$$

where w_d represents words of test document d, M is the topic model, N_d is the number of words in document d.

The perplexity results of LDA with unpooled data and our model are shown in Figure 3. The perplexity of the proposed method is better than LDA without pooling the data. We conducted our experiments using 35 topics, as the improvement in perplexity was low compared to the increase in the runtime. More details are provided in Section 5.7.3.

For topic extraction, we used the MAchine Learning for LanguagE Toolkit (MALLET) [16]. MALLET is a Java based package that implements the LDA model. Table 2 shows an example for top ten words for five topics. After the model is trained, it can be used to predict the topics in unseen corpuses. Thus we can now predict topics distribution for every corpus in our database.

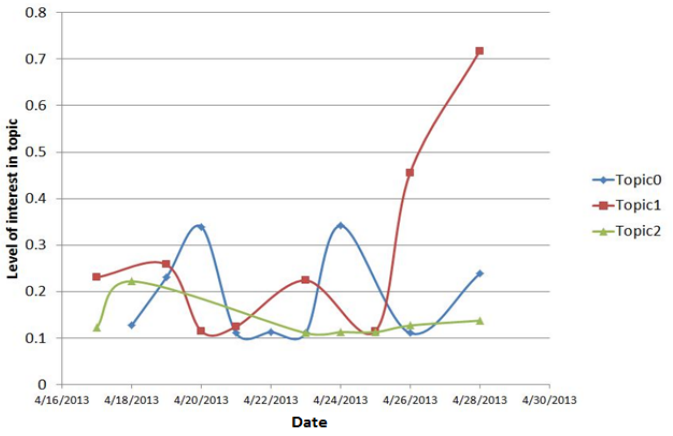

Figure 4: User dynamic level of interest in topics.

5.5 Calculating the Dynamic Level of Interest

In real life, the degrees of popularity of the topics are not constant. There are topics that attracts more users than the others. Also, the user's interest in one topic can change from one time to another. Figure 4 shows an example of one user's changing levels of activity in some of the topics over time. The dynamic level of interest (LoI) is calculated using Equation 2. The user's dynamic LoI is based on the dynamic level of activity of the user in each topic.

5.6 Personalized Recommender Model

Using the features described above, a feature vector was created for all the tweets in the activity windows of the users, as described in Section 4.3. Each of the feature vectors is augmented by a class value. We considered the only possible class values are *interesting* or *not interesting*. The class value is set to be *interesting* if the user replied, retweeted or favored the corresponding tweet. Otherwise, the class value is set to be *not interesting*. We used the feature vectors for each user individually to train three classifiers: *1)* J48, a Java implementation of the C4.5 tree based classifier [20], *2)* supervised Support Vector Machine (SVM), a function based classifier, and *3)* Naive Bayes Classifiers [18]. The three classifiers are used to predict whether the tweets of the timeline is interesting (the user will most likely interact with) or non interesting.

5.7 Discussion

To identify the importance of the *Dynamic LoI* feature compared to the other, we run the experiments twice. First, the runs were done without including the *Dynamic LoI* feature in the training vectors. Thus, this helps to establish a baseline. Then, runs were done with including the *Dynamic LoI* feature to the training vectors. The training and testing of each user data is done separately. We used a 10-fold cross validation scheme to verify our results.

Table 2: Example for top ten words for five topics.

Topic	Top 10 Words
Politics	tcot obama party house gun america vote president romney vote
Technology	app iphone apple google ipad mobile web ios android facebook
Horoscope	libra love capricorn true horoscope virgo cancer money stars sagittarius
Sports	browns game nfl cleveland football coach team eagles win season
Crime	breaking Boston police scene fire sandy shooting victim shot level

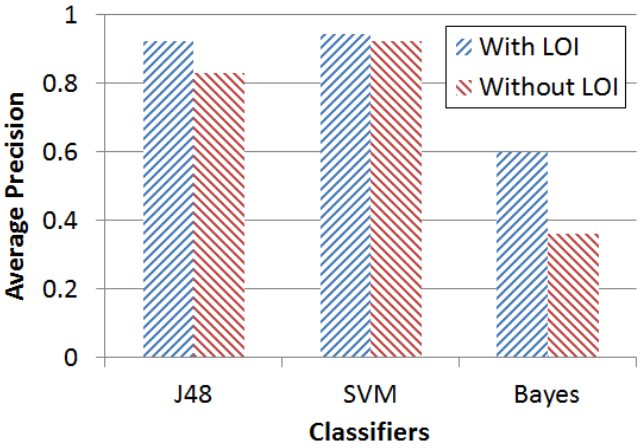

Figure 5: Average precision.

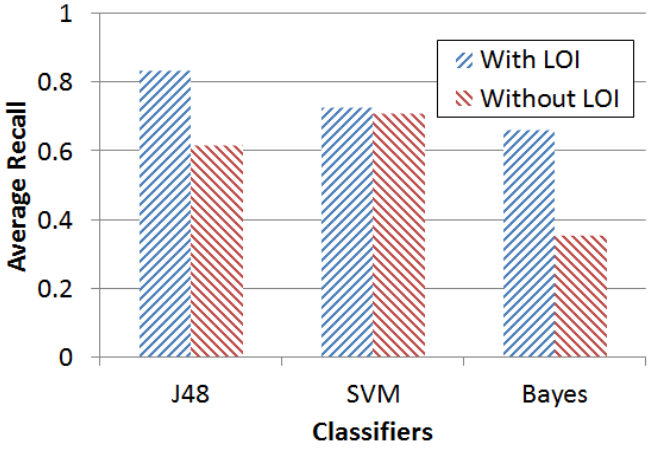

Figure 6: Average recall.

5.7.1 Dynamic LoI and Other Feaures Effect

We recorded two quality measures in our experiments: Precision ($P = TP/(TP + FP)$) and Recall ($R = TP/(TP + FN)$) where TP, FP and FN are the number of true positive, false positive and false negative examples, respectively.

Figure 5 shows the average precision values for the three classifiers. Using the *Dynamic LoI* feature improved the average precision of J48 by 8%, and improved that of the SVM and Naive bayes with about 2% and 36% respectively. It is clear from the figure that the SVM is performing better than J48 and Naive Bayes classifiers, when not relying on the *Dynamic LoI*. The best results is achieved when using *Dynamic LoI* feature with either J48 or SVM classifiers. When using the *Dynamic LoI* feature, the J48 and SVM performed equally. The Naive Bayes performed better with *Dynamic LoI*, but not as good as the other two classifiers.

Figure 6 shows an improvement in J48, SVM and Naive Bayes average recall by 33%, 3% and 80% respectively. J48 is also out performing SVM and the Naive Bayes classifiers when using *Dynamic LoI*.

We had to accurately judge the gain from including the *Dynamic LoI* feature and to determine influence of users who have few tweets, For that we used the concept of 'active users' from the traditional media research [14] and focused on those users with some minimum level of activity. We sorted the users by their activity level in posting, retweeting or favoring the others posts. The users are then divided into three categories according to their activity level (high active, medium active and low active users).

Figures 8, 9 and 10 show the average gain in precision and recall values in J48, Bayes and SVM classifiers, respectively, when including the *Dynamic LoI* feature. A positive value means that including the feature improved the classification, whereas, a negative value means that including the feature worsen the classification. The three figures show that the gain from including the *Dynamic LoI* feature is more when considering users with high activity. This makes our model more important for highly active users. The only negative gain is with the SVM classifier for users with less activity. This is intuitive since less active users do not show their interest in the topics as they do not retweet or reply on the tweets.

We had also evaluated the relative importance of other features used in the classification process. We used Information gain feature selection method to measure the dependence between features and the class labels [10]. Figure 7 shows the features ranked according to their average information gain. It is clear that the LOI feature is considered one of the relatively highest important feature in the classification process as compared to other features.

We analyzed the classification runtime for each classifier. Figure 11 shows the runtime for the three classifiers. It is clear that the SVM has the longest runtime compared to J48 and Bayes classifiers (note the logarithmic scale on the Y-axis).

5.7.2 Tweets Pooling Effect

Since changing the number of words in the documents can greatly affect the output of the topic modeling step, we repeated the experiments after applying the pooling step. We

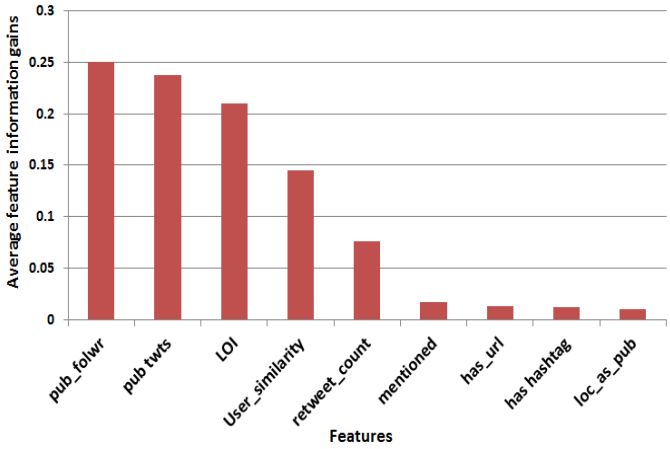

Figure 7: Average feature information gains across all users.

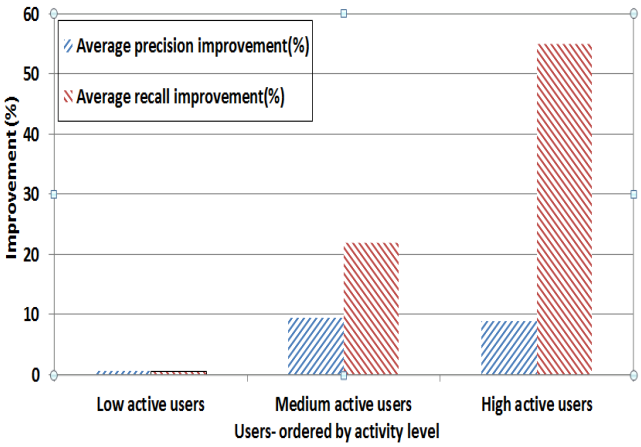

Figure 8: J48 — activity level effect.

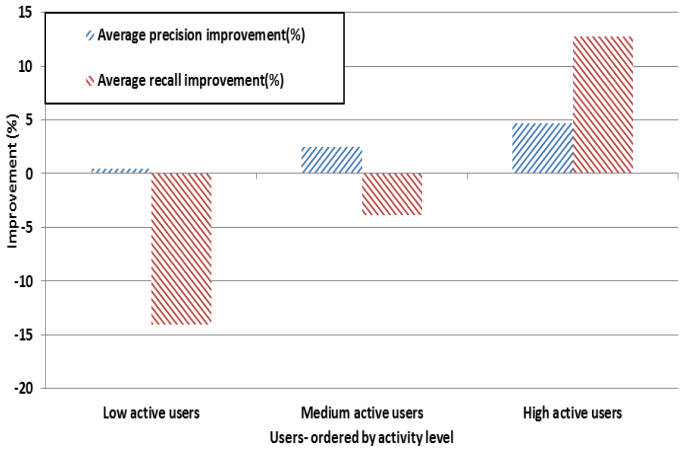

Figure 9: SVM — activity level effect.

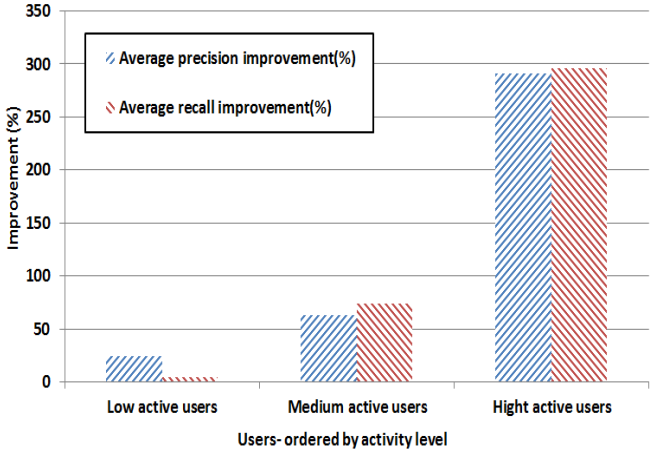

Figure 10: Bayes — activity level effect

.

used the J48 classifier as it had relatively good results with a reasonable run time. Figure 12 shows the pooling effect on improving the precision and the recall. The experiments shows that the average recall was improved by more than 6% without loss in precision.

5.7.3 Number of Topics Variation Effect

Besides our previous experiments, we analyzed the effect of varying the number of topics on the classification process. For example, a user might be interested in a certain topic, but the classifier only recognize the user's interest in a subtopic. We demonstrated this by reconducting the experiments with the J48 classifier while varying the number of topics (Figure 13). Generally, the small number of topics results in very broad topics. This results in poor classification.

On the other hand, large number of topics will result in many very specific topics, as subtopics become the main categories. This also leads to poor classification in our case. Again, the variation of the number of topics has a minor effect on precision, but the recall was improved by 4% by having around 35 topics. The recall value dropped again, when raising the number of topics to 60. So, although the perplexity value was better at 60 topics than 35 topics, the over specification didn't help in our case.

6 Conclusion

In this paper, we introduced the concept of dynamic level of interest (LoI) for microblogs users. To determine the level of interest of the user in a new corpus, we proposed a novel model that is based on topics in that corpus and the history of the user activity in each topic. The goal of the model is to identify the important tweets to a user in his/her timeline.

To illustrate the effectiveness of our model, we used a Twitter APIs to build a dataset with more than five million tweets, and more than 20 thousands users. We demonstrated the importance of using the *Dynamic LoI* feature, by showing the improvement of the average precision and the average recall for the three classifiers used (J48, Naive Bayes, and SVM). Using our approach, we were able to improve the precision and recall of identifying important tweets by up to 36% and 80% respectively. The model analysis showed that the model has higher gain for users with high activity level.

We analyzed the behavior of the LDA topic model to identify the key factors that can affect its performance. We demonstrated that by choosing a proper number of topics

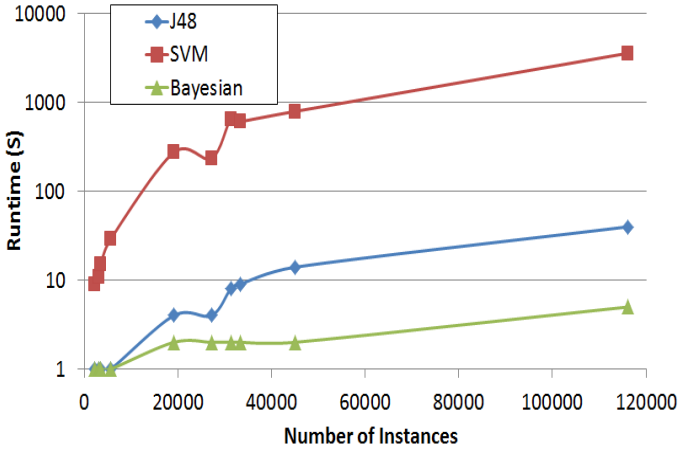

Figure 11: Classification runtime.

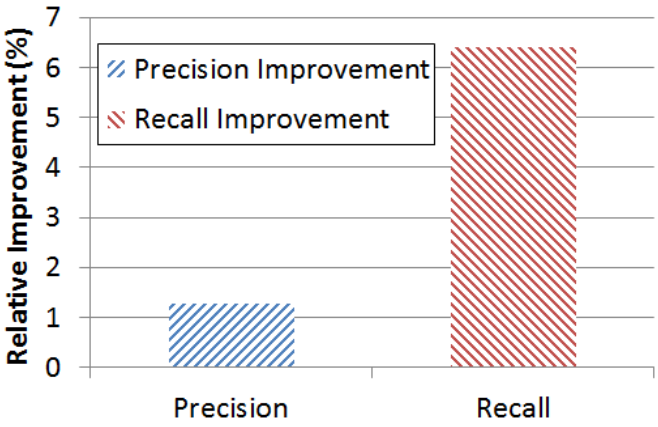

Figure 12: Relative percentage improvement in precision and recall after pooling over J48 with LoI.

and applying pooling techniques to the tweets, an additional 10% improvement can be achieved.

References

[1] About Twitter. `https://about.twitter.com/company`. [Online; accessed 10-September-2014].

[2] Twitter documentation. `https://dev.twitter.com/overview/documentation`. [Online; accessed 15-September-2014].

[3] Twitter homepage. `http://twitter.com`. [Online; accessed 10-November-2014].

[4] Virtual Time Square. `http://vts.cs.vt.edu`. [Online; accessed 10-March-2014].

[5] F. Abel, Q. Gao, G.-J. Houben, and K. Tao. Analyzing user modeling on Twitter for personalized news recommendations. In *Proceedings of the 19th International Conference on User Modeling, Adaption, and Personalization*, UMAP'11, pages 1–12, Berlin, Heidelberg, 2011. Springer-Verlag.

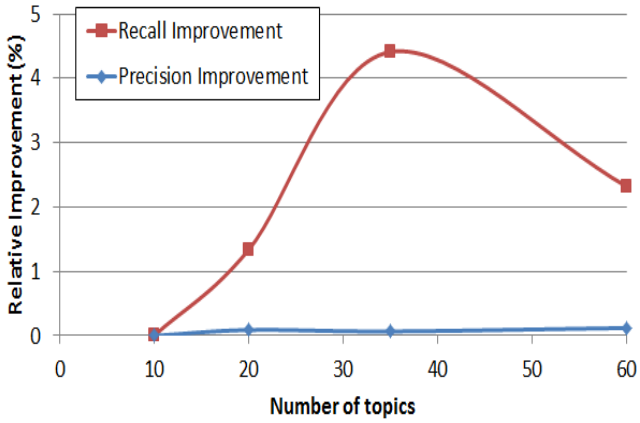

Figure 13: Relative percentage improvement in precision and recall over 10 topics with pooling.

[6] H. Baars and H.-G. Kemper. Management support with structured and unstructured data-an integrated business intelligence framework. *Information Systems Management*, 25(2):132–148, Mar. 2008.

[7] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent Dirichlet allocation. *The Journal of Machine Learning Research*, 3:993–1022, Mar. 2003.

[8] J. Chen, R. Nairn, L. Nelson, M. Bernstein, and E. Chi. Short and tweet: experiments on recommending content from information streams. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '10, pages 1185–1194, New York, 2010. ACM.

[9] K. Chen, T. Chen, G. Zheng, O. Jin, E. Yao, and Y. Yu. Collaborative personalized tweet recommendation. In *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '12, pages 661–670, New York, 2012. ACM.

[10] M. Dash and H. Liu. Feature selection for classification. *Intelligent Data Analysis*, 1(1–4):131–156, 1997.

[11] J. Hannon, M. Bennett, and B. Smyth. Recommending Twitter users to follow using content and collaborative filtering approaches. In *Proceedings of the fourth ACM conference on Recommender systems*, RecSys '10, pages 199–206, New York, 2010. ACM.

[12] H. Kwak, C. Lee, H. Park, and S. Moon. What is Twitter, a social network or a news media? In *Proceedings of the 19th International Conference on World Wide Web*, WWW '10, pages 591–600, New York, 2010. ACM.

[13] D. Laniado and P. Mika. Making sense of Twitter. In *Proceedings of the 9th International Semantic Web Conference on The Semantic Web - Volume Part I*, ISWC'10, pages 470–485, Berlin, Heidelberg, 2010. Springer-Verlag.

[14] M. R. Levy and S. Windahl. The concept of audience activity. In K. E. Rosengren, L. A. Wenner, and P. Palmgreen, editors, *Media gratifications research: Current perspectives*, pages 109–122. Sage Publications, Beverly Hills, CA, 1985.

[15] J. Martinez-Romo and L. Araujo. Detecting malicious tweets in trending topics using a statistical analysis of language. *Expert Systems with Applications*, 40(8):2992–3000, June 2013.

[16] A. K. McCallum. Mallet: A machine learning for language toolkit. http://mallet.cs.umass.edu, 2002.

[17] R. Mehrotra, S. Sanner, W. Buntine, and L. Xie. Improving LDA topic models for microblogs via tweet pooling and automatic labeling. In *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '13, pages 889–892, New York, 2013. ACM.

[18] T. M. Mitchell. *Machine learning*. McGraw Hill series in computer science. McGraw-Hill, Boston, 1997.

[19] M. Pennacchiotti, F. Silvestri, H. Vahabi, and R. Venturini. Making your interests follow you on twitter. In *Proceedings of the 21st ACM International Conference on Information and Knowledge Management*, CIKM '12, pages 165–174, New York, 2012. ACM.

[20] J. R. Quinlan. *C4.5: programs for machine learning*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993.

[21] Z. Ren, S. Liang, E. Meij, and M. de Rijke. Personalized time-aware tweets summarization. In *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '13, pages 513–522, New York, 2013. ACM.

[22] B. Sriram, D. Fuhry, E. Demir, H. Ferhatosmanoglu, and M. Demirbas. Short text classification in Twitter to improve information filtering. In *Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '10, pages 841–842, New York, 2010. ACM.

[23] M. Steyvers, P. Smyth, M. Rosen-Zvi, and T. Griffiths. Probabilistic author-topic models for information discovery. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '04, pages 306–315, New York, 2004. ACM.

[24] I. Uysal and W. B. Croft. User oriented tweet ranking: a filtering approach to microblogs. In *Proceedings of the 20th ACM international conference on Information and knowledge management*, CIKM '11, pages 2261–2264, New York, 2011. ACM.

[25] H. M. Wallach, I. Murray, R. Salakhutdinov, and D. Mimno. Evaluation methods for topic models. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pages 1105–1112, New York, 2009. ACM.

[26] J. Weng, E.-P. Lim, J. Jiang, and Q. He. Twitterrank: Finding topic-sensitive influential twitterers. In *Proceedings of the Third ACM International Conference on Web Search and Data Mining*, WSDM '10, pages 261–270, New York, 2010. ACM.