Recommendation System: Bootstrapping New Tweeter Users

Gregorios A Katsios[1] and Akshit Bansal[1]

[1]Department of Computer Science, SUNY at Albany

Abstract

In this work, we apply unsupervised learning techniques and build a Twitter recommendation system that suggests to newcomers to the social media platform, other users to follow. Our experimental system makes use of external documents the supposedly new user provides in order to generate a topical profile. This topical profile can then be compared for similarities against the topical profiles of previous Tweeter users and therefore make recommendations. We evaluate our techniques against a subset of Tweeter users and their tweets to demonstrate their potential to kick-start the recommendation system and make meaningful recommendations. Our technique surpasses other baselines by 0.0031 in terms of precision and by 0.0026 in terms of recall.

1 Introduction

Real-time social media, such as Twitter and Facebook, are at the center of recommendation systems research. In this project, we focus on Twitter, a simple social media platform that allows users to post 140-character long messages (*tweets*) to a continuously updating open-to-view timeline of user messages [KLPM10]. Users can follow other users in order to receive notifications regarding their posts, and thus effectively creating a filtered view of the time-line of these people. To receive the most gain from the Twitter platform, users carefully select other people to follow, so that they can benefit from their tweets and minimize the risk of missing an interesting update.

As soon as a new account is registered with Twitter, it's owner is asked to select a few existing users to follow and a number of preferred topics from a series of suggestions. With this set-up procedure, Twitter collects some information from the new user in order to build his or her profile, as well as to utilize it towards future recommendations. However, Twitter recommendation systems seem unsuccessful at reliably predicting user accounts that a new user may want to follow. Suffering from the cold start problem (explained in [LKH14] and [ZLZZ10]), recommendation systems cannot form a thorough initial judgment about a new user. In our work, we attempt to mend this weakness and propose a novel approach of bootstrapping a new user. With this method, upon the creation of an account, the new user will be asked to also provide a few links to articles or blogs that he or she enjoys reading. This additional input could possess the ability to help new users connect to other people quicker, satisfying their need to become part of a social group.

The hypothesis is that these documents can be used to create an introductory topical profile for the user as a means to suggest a list of people to follow. To answer this question, we utilize content-based recommendation strategies, described in [VJO$^+$11] and [HBS10], as they rely on the content of tweets and can be easily extended to other sources of textual input, such as articles or blogs. The main assumption is that users with similar interests will be more likely to follow each other, where similarity is measured by comparing the topical profiles of two users. The topical profiling and recommendation algorithms that we use are discussed in detail, as well as the description of the evaluation method of these techniques. For the evaluation, we make use of real-world data collected from 7,988 Twitter users in a standard off-line evaluation that compares the relative performance of our recommendation technique. The results of this evaluation are also discussed and presented.

2 Related Work

In [HBS10], Hannon el al. illustrate a system called "Twittomender", which recommends to a user who to follow using a content-based approach. Here, each user is represented by either their own tweets, their followees' tweets, their followers' tweets, or a combination of everything. In the case where a user is represented by his own tweets, users with similar tweets are recommended to the targeted user. In the case where a user is represented by their followees' tweets, a target user is recommended with a list of users whose followees' tweets are similar to those of this user's followees. The remaining two cases are also treated analogously. What all of these cases have in common is that each user is represented by TF-IDF weighting scheme. TF-IDF is short for *term frequency–inverse document frequency*, and it is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus. Similar to this, in our work we represent each user by their own tweets, following the TF-IDF weighting scheme. However, since we assume the users in the testing set (as explained in Section 3.1) have not tweeted anything yet, we use the provided external documents instead.

In [VJO+11], the authors present a collection of recommendation techniques and implement a testing environment to analyze their advantages and disadvantages. The authors also explain how the small world theory, the idea that people who are closer to an individual have more influence over this individual's opinions, affects recommendations. From this work, we adopt the proposed k-nearest neighbor algorithm, which is based on a similarity concept to build a group of object to return as recommendations, where in our case the objects of recommendation are other users.

Both [ZLZZ10] and [LKH14] describe the cold-start problem as being related to recommendations for new users or items, where the system does not have information about their preferences in order to make recommendations. Zang et al. focus their research on user-tag-object tripartite graphs, and propose a recommendation algorithm that considers the frequencies of tags as user preferences on different topics and tag-object links as semantic relations between them. On the other hand, Lika et al. propose a mechanism that takes into consideration demographic information by building user neighborhoods based on how similar the users are. The authors of [LKH14] make the assumption that people with common background and similar characteristics have more possibilities to have similar preferences, which is analogous to our main assumption and to the ideas discussed in [VJO+11].

[GGL+13] explores the architecture of the recommendation system currently in use by Twitter, "Who To Follow". The authors discuss about the practical aspects of building and running a large-scale recommendation service, and stress that the success of a service depends on many factors besides the algorithm itself. One such example is having the right architecture and the right evaluation framework for iterative refinement. Even though we do not consider running our experimental system live on Twitter, the k-nearest neighbor algorithm that we are using can be easily extended and deployed as a service on the platform.

A comprehensive survey regarding recommendation systems for Twitter is presented in [KLZ12]. The authors attempt to categorize the existing works in the field, by using a taxonomy to classify different recommendation tasks. This process has revealed areas that have received very little attention, such as hashtag and retweet recommendations. While the task of recommending who to follow is well studied, only a modest amount of work has been made using content-based approaches that utilize users' tweets in order to perform recommendations.

3 Method

In this section, we will describe the data that we have collected and the details to our methods. Our experimental framework makes use of a processing pipeline in order to provide recommendation to target users. First step is to prepare the topical profile of each user, which is a numerical vector composed of TF-IDF values. As second step, the vector of a target user is fed into the k-nearest neighbor algorithm which compares it with all of the existing users' vectors through a similarity metric and returns the k most similar users. Finally, these most similar users are suggested to the target user as people he or she might want to follow.

3.1 Data Acquisition

The Twitter user profiles we use during our evaluation were downloaded using Tweepy[1], which is a module for the Python programming language. Tweepy interfaces with the Twitter API and allows developers to seamlessly communicate with the platform.

[1] `http://docs.tweepy.org/en/v3.5.0/index.html`

	Users	Tweets	URLs
Total	7988	364475	88934
Well Connected	945	49344	10817
Test Set	798	42130	10054

Table 1: High-level information regarding the dataset.

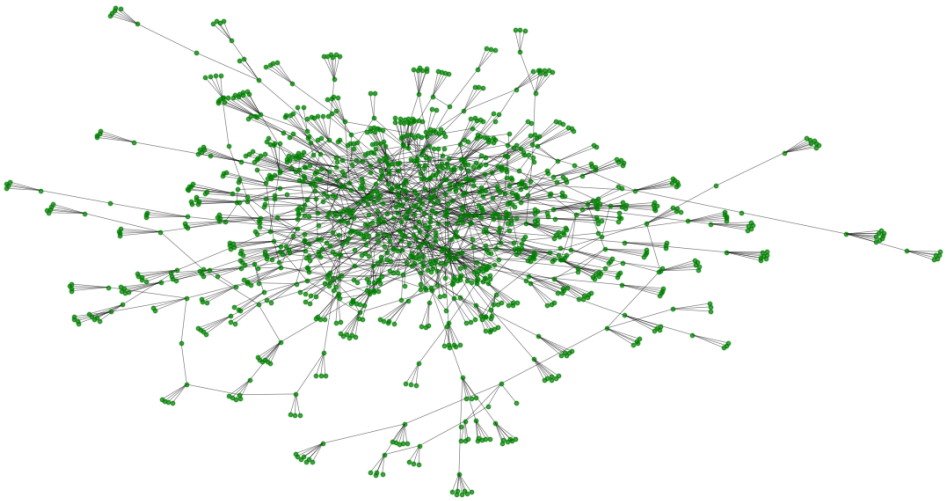

Figure 1: Social graph of the *well connected* users.

Our strategy for collecting users adheres to the common snowball sampling technique. In snowball sampling, the data collection process begins with a single seed user and subsequent users are added by referring to his or her follower list. The process is repeated for the newly added users until we reach the desired amount of data. The main advantage of this methodology is that it can locate users of a specific population, without using a predefined membership list. In our case, the initial seed user was "TheRealStanLee"[2]. For each sampled user, we download their 50 most recent tweets and we limit the number of extracted followers to 20. We add a user to our dataset only if the tweets are written in English.

In order to evaluate our bootstrapping technique, we would require a testing set composed of users who follow at least 5 other users who also exist in our dataset. From the 7,988 Twitter user profiles that we collected, we select the 798 such users and designate them to be the testing set. In the context of our evaluation, we assume that these users are newcomers to the social media platform and have not made any posts yet. Furthermore, since we cannot directly ask for links to articles or blogs that they enjoy reading, for each user in the testing set we arbitrarily pick 10 URLs that appear in their recent tweet history, if any and extract their textual content. Table 1 indicates the number of users in our dataset as well as how many tweets and URLs these users have posted in their recent timeline. Classified as *well connected* users are the ones that had 5 or more followers or friends, and in Figure 1 we can see their social graph including their followers.

[2] `https://twitter.com/TheRealStanLee`

Rank	Term	Count
1	People	976065
2	Time	892175
3	Years	646713
4	Way	574182
5	Year	439762
6	World	407473
7	Day	404900

Table 2: Top-7 most common nouns.

Rank	Term	Count
1	Trump	14423
2	McCabe	6718
3	Gun	4333
4	Mueller	3513
5	Democrat	3341
6	FBI	3307
7	Love	3272

Table 3: Top-7 most tweeted topics.

3.2 Topical Profiling

Before we can have our system perform recommendations, we must generate the topical profiles for all users. Since each user is represented by his or her own tweets, for each user U_i in our dataset we construct a list of words $L_i = [w_1, w_2, \cdots, w_N]$. The elements of L_i are the lemmas of the nouns found in user's U_i tweets, as nouns are good topic indicators. In English, the lemma of a noun is its singular form: e.g., waiter rather than waiters. For grammatical reasons, while tweeting, users are going to use different forms of a word. The goal of lemmatization is to reduce inflectional forms and sometimes derivationally related forms of a word to a common base form [SMR08]. To get the base forms we use the lemmatizer provided in NLTK[3], which is a module for the Python programming language.

Having a list of nouns each user uses, we build a dictionary that maps each term to the number of times it was used across all users. For the purpose of the evaluation, when calculating the term frequency we do not count instances of nouns that were tweeted by a user reserved for testing. From this process, we also omit certain nouns that appear very commonly in English text and discourse, and thus do not represent proper discussion topics. The exclusion list contains 100 common nouns and it was constructed by using the Corpus of Contemporary American English [Dav08]. Table 2 shows the 7 most frequent common nouns; words such as *time*, *year*, *people*, *way*, etc. do not offer themselves as coherent topics of conversation. The reasoning of constructing this dictionary is to identify the most popular topics discussed by the users in our dataset; the 7 most tweeted topics are listed in Table 3.

[3] `https://www.nltk.org/`

3.2.1 TF-IDF Representation

From the dictionary discussed in the above paragraphs, we extract the 100 most frequently mentioned topics and designate them as our *feature set* F, where $F = \{t_1, t_2, \cdots, t_{100}\}$. After that, we calculate the TF-IDF values for all terms in the feature set with respect to each user. The TF-IDF is the product of two statistics, term frequency and inverse document frequency.

In the case of the term frequency $tf(t_j, L_i)$ we use the number of times that term t_j appears in the tweets of user U_i by exploiting the pre-calculated L_i. If we denote this raw count by f_{t_j, L_i}, then we have:

$$tf(t_j, L_i) = f_{t_j, L_i}$$

The inverse document frequency is a measure of how important a word is, meaning how often we can find a user that has tweeted the word. We calculate $idf(t_j)$ as the logarithm of the fraction of the the total number of users to the number of users that have used the term in any of their tweets.

$$idf(t_j) = log\Big(\frac{U_{Total}}{|\{U_i : t_j \in L_i\}|}\Big)$$

With:

- U_{Total}: the total number of users.
- $|\{U_i : t_j \in L_i\}|$: the number of users where the term t_j appears in any of their tweets, i.e. $tf(t_j, L_i) \neq 0$.

Thus, each user U_i is assigned a 100-dimensional vector $T_i = [v_1, v_2, \cdots, v_{100}]$ as his or her topical profile, where:

$$\begin{aligned} v_j &= tfidf(t_j, L_i) \\ &= tf(t_j, L_i)\ idf(t_j) \end{aligned}$$

3.3 *k*-Nearest Neighbors

The k-nearest neighbor algorithm (k-NN), originally proposed in [CH+74], utilizes a similarity metric to build a group of objects (the nearest users) from where candidates for recommendation are drawn. Each user is classified based on its similarity to the target user, for whom the prediction is being made. The list of the most similar users is identified by selecting the top-k most similar users.

The user based recommendation algorithm (Return list R_t of top-k users, sorted by similarity in descending order) can be described as the process of recommending to target user U_t other users to follow as:

Algorithm 1: k-NN Recommender.

Input : target user U_t, integer k
Output: list R_t containing top-k similar users
`/* initialize epmty list` R_t `*/`
$R_t = [\,]$;
T_t = getTopicalProfile(U_t);
for *each user* $U_i \neq U_t$ **do**
  T_i = getTopicalProfile(U_i);
  cos = $sim(T_t, T_i)$;
  `/* append similarity to list` R_t `*/`
  R_t.append([U_i, cos]);
end
`/* sort` R_t `by descending order of cos */`
R_t.sort();
`/* keep` k `first entries */`
$R_t = R_t[:k]$;

3.3.1 Similarity Metric

The Algorithm 1 of k-NN described above makes use of the similarity metric $sim(T_t, T_i)$. We define $sim(T_t, T_i)$ as the cosine similarity between the target user U_t and any other user U_i as the dot product of their topical profile vectors T_t and T_i divided by their magnitude:

$$\begin{aligned} sim(T_t, T_i) &= cos(\theta) \\ &= \frac{T_t \cdot T_i}{||T_t||\ ||T_i||} \\ &= \frac{\sum_{p=1}^{100} T_{t_p} T_{i_p}}{\sqrt{\sum_{p=1}^{100} T_{t_p}^2}\sqrt{\sum_{p=1}^{100} T_{i_p}^2}} \end{aligned}$$

Where:

- T_{t_p} and T_{i_p} are components of vector T_t and T_i respectively.

The cosine similarity of two documents will range from 0 to 1, since the TF-IDF weights cannot be negative. A $sim(T_t, T_i) = 1$ means that the two profiles are exactly the same and $sim(T_t, T_i) = 0$ indicates orthogonality. In-between values indicate intermediate similarity. Cases that compare zero-vectors are interpreted as zero value evaluations [RIS+94].

4 Experimental Results

To evaluate our recommendation techniques, we make use of real-world data collected from 7,988 Twitter users in a standard off-line evaluation and compare their relative performance. As explained in Section 3.1, about 10% of the users were reserved as our testing set.

4.1 Evaluation

For these reserved users, we apply lemmatization (as explained in Section 3.2) on the textual content obtained by following their URLs, thus adding to them the list $L_{i(url)}$. Furthermore, we apply the TF-IDF calculation of the same feature set detailed in Section 3.2.1 with regards to $L_{i(url)}$. This produces a 100-dimensional vector $T_{i(url)} = [v_1, v_2, \cdots, v_{100}]$, which will function as the introductory topical profile of a supposedly new user. For the recommendation process, since we attempt to measure how well our bootstrapping technique performs, we draw target users from our testing set. Thus, every target user U_t can be represented by his or her $T_{t(url)}$ vector in the similarity function. For each user U_t in the test set we use their $T_{t(url)}$ to find the list R_t of the k-nearest neighbors out of the entire dataset of all users.

Then we can compute the precision $prc(U_t)$ and recall $rec(U_t)$ of the recommendation for target user U_t, by comparing list R_t with the list A_t of the actual people that he or she follows:

$$prc(U_t) = \frac{\#TP}{\#TP + \#FP} \quad rec(U_t) = \frac{\#TP}{\#TP + \#FN}$$

Where $\#TP$, $\#FP$ and $\#FN$ are the number of true positive, false positive and false negative in-

stances respectively. We calculate these as:

$$\begin{aligned} \#TP &= |\{U_i | U_i \in R_t \text{ and } \in A_t\}| \\ &= |A_t \cap R_t| \end{aligned}$$

$$\begin{aligned} \#FP &= |\{U_i \in R_t | U_i \notin A_t\}| \\ &= |R_t \setminus A_t| \end{aligned}$$

$$\begin{aligned} \#FN &= |\{U_i \in A_t | U_i \notin R_t\}| \\ &= |A_t \setminus R_t| \end{aligned}$$

In order to compare the performance of our technique against other baselines, we evaluate the recommendation system on three different settings: First, as though the new users provide links to articles or blogs that they enjoy reading and we construct their introductory topical profile $T_{i(url)}$ by using these documents. Second, as though the new users do not provide additional links, but select a few hashtags during their account creation so we construct their introductory topical profile $T_{i(hashtag)}$; this intends to approximate Twitter's current account set-up procedure. $T_{i(hashtag)}$ is the topical profile that was generated by only considering the hashtags user U_i has used in their tweets. Third, as though the new users have posted a number of tweets in their timeline and we are able to construct their regular topical profile T_i.

4.2 Results

Our basic measures of recommendation performance are mean precision and mean recall. Precision is the fraction of the correctly suggested users (present in the target user's actual followees-list) over the total number of suggested users. Recall is the fraction of correctly suggested users over the total number of users present in the target user's actual followees-list. To begin with, Figures 2 and 3 graph the average precision and average recall respectively, versus various recommendation-list sizes for the three different evaluation settings, using the 798 user test set.

Overall the different recommendation strategies appear to perform poorly across all different implemented methods. This phenomenon can be explained due to the small size of our dataset and the restraints we imposed during collection, such as retrieving only 20 followers per user to favor collection speed. This means that two users could be present in our dataset

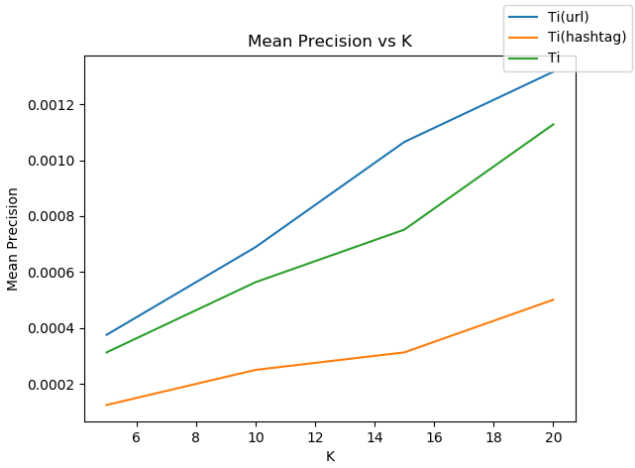

Figure 2: Mean precision vs. k for the three settings.

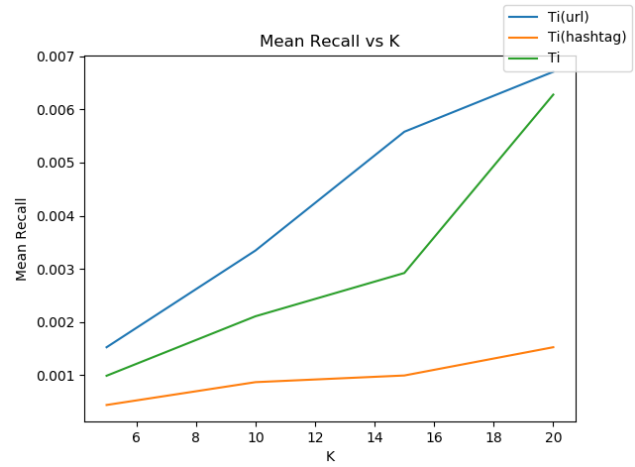

Figure 3: Mean recall vs. k for the three settings.

and seem unrelated, but in Twitter they may be following each other. In fact, the recommendations proposed by our system may be correct, but we do not have the ground truth to properly compare with.

Interestingly though, we can observe that even with this limited dataset, our bootstrapping technique seems to outperform all other baselines. For $k = 15$, it surpasses the strategy where target users are represented by their actual tweets by 0.00031 in terms of precision and by 0.0026 in terms of recall. In addition, against the strategy where target users are represented by their hashtags, the improvement is clear for all values of k, with an improvement of at least 0.00025

and 0.001 in terms of precision and recall respectively. This improvement is a direct result from the fact that articles and blogs provide significantly more terms to build a topical profile from; these documents are not restricted to 140-character long messages, as Twitter messages are.

5 Discussion

In this project, with the use of unsupervised learning techniques, we build a Twitter content-based recommendation system that suggests to newcomers to the social media platform, other users to follow. We demonstrate that collecting additional information about a users preferences during the account creation process, helps ameliorate the cold-start problem of recommendation systems.

In this scenario, we utilize external documents such as blogs or articles that have been provided by the new user during account set-up to build their introductory topical profile. We define a topical profile to be a numerical vector of TF-IDF values, where its dimensions are derived from a list of popular topics. Then we compare these vectors with the ones generated from the tweets of existing Twitter users and suggest a list of users to follow by taking the top-k most similar users.

We evaluate our technique against other baselines to demonstrate the potential to kick-start the recommendation system and make meaningful recommendations. Our bootstrapping technique surpasses the other two methods by 0.0031 in terms of precision and by 0.0026 in terms of recall. Moreover, it shows potential to outperform recommendation systems that represent users by their own tweets only.

The natural continuation of this project would be to verify our current results on state-of-the-art recommendation systems, where matrix factorization approaches can be exploited. Matrix factorization techniques discover latent features underlying the interactions between users and thus produce more accurate recommendations [KBV09].

References

[CH+74] TM Cover, PE Hart, et al. Nearest neighbor classifiers. *IEEE Transactions on Computers*, pages 23–11, 1974.

[Dav08] Mark Davies. *The corpus of contemporary American English*. BYE, Brigham Young University, 2008.

[GGL+13] Pankaj Gupta, Ashish Goel, Jimmy Lin, Aneesh Sharma, Dong Wang, and Reza Zadeh. Wtf: The who to follow service at twitter. In *Proceedings of the 22nd international conference on World Wide Web*, pages 505–514. ACM, 2013.

[HBS10] John Hannon, Mike Bennett, and Barry Smyth. Recommending twitter users to follow using content and collaborative filtering approaches. In *Proceedings of the fourth ACM conference on Recommender systems*, pages 199–206. ACM, 2010.

[KBV09] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8), 2009.

[KLPM10] Haewoon Kwak, Changhyun Lee, Hosung Park, and Sue Moon. What is twitter, a social network or a news media? In *Proceedings of the 19th international conference on World wide web*, pages 591–600. ACM, 2010.

[KLZ12] Su Mon Kywe, Ee-Peng Lim, and Feida Zhu. A survey of recommender systems in twitter. In *International Conference on Social Informatics*, pages 420–433. Springer, 2012.

[LKH14] Blerina Lika, Kostas Kolomvatsos, and Stathes Hadjiefthymiades. Facing the cold start problem in recommender systems. *Expert Systems with Applications*, 41(4):2065–2073, 2014.

[RIS+94] Paul Resnick, Neophytos Iacovou, Mitesh Suchak, Peter Bergstrom, and John Riedl. Grouplens: an open architecture for collaborative filtering of netnews. In *Proceedings of the 1994 ACM conference on Computer supported cooperative work*, pages 175–186. ACM, 1994.

[SMR08] Hinrich Schütze, Christopher D Manning, and Prabhakar Raghavan. *Introduction to information retrieval*, volume 39. Cambridge University Press, 2008.

[VJO+11] B Valois Jr, Marcius Armada de Oliveira, et al. Recommender systems in social networks. *JISTEM-Journal of Information Systems and Technology Management*, 8(3):681–716, 2011.

[ZLZZ10] Zi-Ke Zhang, Chuang Liu, Yi-Cheng Zhang, and Tao Zhou. Solving the cold-start problem in recommender systems with social tags. *EPL (Europhysics Letters)*, 92(2):28002, 2010.