Empirical Study of Topic Modeling in Twitter

Liangjie Hong and Brian D. Davison
Dept. of Computer Science and Engineering, Lehigh University
Bethlehem, PA 18015 USA
{lih307,davison}@cse.lehigh.edu

ABSTRACT

Social networks such as Facebook, LinkedIn, and Twitter have been a crucial source of information for a wide spectrum of users. In Twitter, popular information that is deemed important by the community propagates through the network. Studying the characteristics of content in the messages becomes important for a number of tasks, such as breaking news detection, personalized message recommendation, friends recommendation, sentiment analysis and others. While many researchers wish to use standard text mining tools to understand messages on Twitter, the restricted length of those messages prevents them from being employed to their full potential.

We address the problem of using standard topic models in micro-blogging environments by studying how the models can be trained on the dataset. We propose several schemes to train a standard topic model and compare their quality and effectiveness through a set of carefully designed experiments from both qualitative and quantitative perspectives. We show that by training a topic model on aggregated messages we can obtain a higher quality of learned model which results in significantly better performance in two real-world classification problems. We also discuss how the state-of-the-art Author-Topic model fails to model hierarchical relationships between entities in Social Media.

Categories and Subject Descriptors

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

General Terms

Algorithms, Design, Experimentation

Keywords

Twitter, Topic Models, Social Media

1. INTRODUCTION

In recent years, social networks such as Facebook, Myspace and Twitter have become important communication tools for people across the globe. These websites are increasingly used for communicating breaking news, eyewitness accounts and organizing large groups of people. Users of these websites have become accustomed to receiving timely updates on important events, both of personal and global value. For instance, Twitter was used to propagate information in real-time in many crisis situations such as the aftermath of the Iran election, the tsunami in Samoa and the Haiti earthquakes. Many organizations and celebrities use their Twitter accounts to connect to customers and fans.

Recent studies in a variety of research areas show increasing interests in micro-blogging services, especially Twitter. Early work mainly focused on quantitative studies on a number of aspects and characteristics of Twitter. For example, Java et al. [6] studied the topological and geographical properties of Twitter's social network in 2007 and found that the network has high degree correlation and reciprocity, indicating close mutual acquaintances among users. Krishnamurthy et al. [7] studied the geographical distribution of Twitter users and their behaviors among several independent crawls. The authors mostly agree with the classification of user intentions presented by Java et al., but also point out evangelists and miscreants (spammers) that are looking to follow anyone. Weng et al. [19] studied the problem of identifying influential users on Twitter by proposing an extension of the PageRank algorithm to measure the influence taking both the topical similarity between users and the link structure into account. They also presented evidence to support the existence of homophily in Twitter. In their work, they utilized topic models (described below) to understand users' interests.

Among the research mentioned above and others, researchers wish to use messages posted by users to infer users' interests, model social relationships, track news stories and identify emerging topics. However, several natural limitations of messages prevent some standard text mining tools to be employed with their full potentials. First, messages on Twitter (which are called "tweets") are restricted to 140 characters. This is substantially different from traditional information retrieval and web search. Second, within this short length, users invented many techniques to expand the semantics that are carried out by the messages. For example, when posting external URLs, users may use URL shortening services (e.g., http://www.bit.ly). In addition, users heavily use self-defined hash tags starting with "#" to identify certain events or topics. Therefore, from the perspective of length (e.g., in characters), the content in messages is limited while it may convey rich meanings.

Topic models [1] are powerful tools to identify latent text patterns in the content. They are applied in a wide range of areas including recent work on Twitter (e.g., [13]). Social media differs from some standard text domain (e.g., citation network, web pages) where topic models are usually utilized in a number of ways. One important fact is that there exists many "aggregation strategies" in social media that we usually want to consider them simultaneously. For example, on Twitter, we usually want to obtain topics associated with messages and their authors as well. Researchers typically only discuss one of them. Weng et al. [19] trained a topic model on aggregated users' messages while Ramage et al. [13] used a slightly modified topic model on individual messages. Neither of them mentioned the other possibility. Indeed, to our knowledge, there is no empirical or theoretical study to show which method is more effective, or whether there exists some more powerful way to train the models.

In this paper, we want to address the problem of how to effectively train a standard topic model in short text environments. Although our experiments are solely based on Twitter, we believe that some of the discussions can be also applied to other scenarios, such as chat logs, discussion boards and blog comments. More specifically, we want to answer these questions in the paper:

- If we use different aggregation strategies and train topic models, do we obtain similar topics or are the topics learned substantially different?
- Can we learn a topic model more quickly that retains its usefulness, without any modifications to standard models?
- Can we shed some light on how we can build new models to fully utilize the structure of short text environments?

With a set of carefully designed experiments in both quantitative and qualitative perspective and two more real-world classification problems, in this paper, we make the following contributions:

- Topics learned by using different aggregation strategies of the data are substantially different from each other.
- Training a standard topic model on aggregated user messages leads to a faster training process and better quality.
- Topic mixture distributions learned by topic models can be a good set of supplementary features in classification problems, significantly improving overall classification performance.

The paper is organized as follows. In Section 2, we outline some related work on the topic. In Section 3, we introduce several methods to learn topic models on Twitter. Section 4 details our experiments and major conclusions. In Section 5, we summarize our contributions.

2. RELATED WORK

Topic modeling is gaining increasingly attention in different text mining communities. Latent Dirichlet Allocation (LDA) [3] is becoming a standard tool in topic modeling. As a result, LDA has been extended in a variety of ways, and in particular for social networks and social media, a number of extensions to LDA have been proposed. For example, Chang et al. [4] proposed a novel probabilistic topic model to analyze text corpora and infer descriptions of the entities and of relationships between those entities on Wikipedia. McCallum et al. [10] proposed a model to simultaneously discover groups among the entities and topics among the corresponding text. Zhang et al. [22] introduced a model to incorporate LDA into a community detection process. Similar work can be found in [8] and [11]

Related to this work, where we need to obtain topic mixture for both messages and authors, Rosen-Zvi et al. [16] introduced an author-topic model, which can flexibly model authors and their corresponding topic distributions. In their experiments, they found that the model outperforms LDA when only small number of words are observed in the test documents. Ramage et al. [14, 13] extended LDA to a supervised form and studied its application in micro-blogging environment. Phan et al. [12] studied the problem of modeling short text through LDA. However, their work mainly focused on how to apply it to Wikipedia and they did not provide any discussion on if there is other ways to train a same model.

In web search, this line of research usually employs search engines directly. For example, Sahami et al. [17] introduced a kernel function based on search engine results. Yih et al. [21] further extended the method by exploiting some machine learning techniques.

3. METHODOLOGY

In this section, we will introduce several methods to train topic models on Twitter and discuss their technical details. In this paper, we mainly consider two basic models: LDA and author-topic model [15]. We first briefly review these two models and then discuss their adaptation to Twitter.

3.1 LDA and the Author-Topic Model

Latent Dirichlet Allocation is an unsupervised machine learning technique which identifies latent topic information in large document collections. It uses a "bag of words" approach, which treats each document as a vector of word counts. Each document is represented as a probability distribution over some topics, while each topic is represented as a probability distribution over a number of words. LDA defines the following generative process for each document in the collection:

1. For each document, pick a topic from its distribution over topics.
2. Sample a word from the distribution over the words associated with the chosen topic.
3. The process is repeated for all the words in the document.

More formally, each document in the collection is associated with a multinomial distribution over T topics, which is denoted as θ. Each topic is associated with a multinomial distribution over words, denoted as ϕ. Both θ and ϕ have Dirichlet prior with hyper-parameters α and β respectively. For each word in one document d, a topic z is sampled from the multinomial distribution θ associated with the document and a word w from the multinomial distribution ϕ associated with topic z is sampled consequently. This generative process is repeated N_d times where N_d is the total number of words in the document d.

The Author-Topic Model (AT model) is an extension of LDA, which was first proposed in [16] and further expanded in [15]. Under this model, each word w in a document is associated with two latent variables: an author, x and a topic, z. Similarly to LDA, each author in the collection is associated with a multinomial distribution over T topics, denoted as θ. Each topic is associated with a multinomial distribution over words, denoted as ϕ. Here, differing from LDA, the observed variables for an individual document is the set of authors and the words in the document. The formal generative process of Author-Topic Model is as follows:

1. For each document, given the vector of authors.
2. For each word in the document, conditioned on the author set a_d, choose an author $x_{di} \sim \text{Uniform}(a_d)$.

3. Conditioned on x_{di}, choose a topic z_{di}.

4. Conditioned on z_{di}, choose a word w_{di}.

Here, one important difference between the AT model and LDA is that there is no topic mixture for an individual document. Therefore, if we want to model documents and authors simultaneously, certain extension or special treatment is needed. A detailed description of the model can be found in [15].

3.2 Topic Modeling Schemes

Recall that our goal is to infer a topic mixture θ for both messages and authors in the corpus. In this sub-section, we will introduce several methods to achieve this goal.

First, we discuss a very natural choice of training models. The process is as follows:

1. Train LDA on all training messages.

2. Aggregate all training messages generated by the same user into a training profile for that user.

3. Aggregate all testing messages generated by the same user into a testing profile for that user.

4. Taking training user profiles, testing user profiles and testing messages as "new documents", use the trained model to infer a topic mixtures for each of them.

We denote this method as the MSG scheme. Note that we do not combine all user profiles into a single set of user profiles simply because some users may be part of the training set, and thus the aggregation of all user profiles may give an unfair advantage to the model to achieve better performance.

We can also train the model on aggregated user profiles, which leads to the following process:

1. Train LDA on aggregated user profiles, each of which combines all training messages generated by the same user.

2. Aggregate all testing messages generated by the same user into testing user profiles.

3. Taking training messages, testing user profiles and testing messages as "new documents", use the trained model to infer a topic mixture for each of them.

We denote the method as the USER scheme.

The third scheme, which we denote as the TERM scheme, is more unusual. The process is as follows:

1. For each term in the training set, aggregate all the messages that contain this term into a training term profile

2. Train LDA on all training term profiles.

3. Build user profiles in training and testing set respectively.

4. Taking training messages, training user profiles, testing user profiles and testing messages as "new documents", use the trained model to infer a topic mixture for each of them.

The rationale for this scheme is that on Twitter, users often use self-defined hash tags (i.e., terms starting with "#") to identify certain topics or events. Building term profiles may allow us to obtain topics related to these hash tags directly.

These schemes each have their own advantages. For MSG, it is straightforward and easily understandable but the training process is based on individual messages, whose content is very limited. The model may not have enough information to learn the topic patterns. More specifically, the occurrences of terms in one message play less discriminative role compared to lengthy documents (e.g., aggregated user profiles or term profiles) where the model has enough term counts to know how terms are related. For the USER and TERM schemes, the models have enough content and might provide a more "accurate" result.

For the AT model, we extend it to allow each message to have a "fictitious" author who is indeed the message itself. Thus, for each message, we either sample the words from the author specific topic mixture or sample them from the message specific topic mixture. Note that the relationship between message specific "route" and author specific "route" is "OR". In other words, we can imagine the process is that an author is writing a message that he will mainly choose the words he is usually interested while choosing some set of words more specific to the current message. Therefore, under this assumption, most of terms in a particular message will choose author "route". This "OR" relationship indeed allows us learn a relatively accurate model for authors but less satisfied model for messages. In our experiments, we find that the topic mixture for messages learned by the extended AT model is usually too sparse and leads to worse results than the MSG scheme. In this paper, we use the AT model to denote the extended AT model with message specific mixtures.

There is another aspect of issues related to different schemes. Usually, the number of users is several magnitude less than the number of messages. Therefore, it would take significantly less time to train a model with the USER scheme rather than the MSG scheme. The same argument can be made for the TERM scheme as well. In addition, the assumption of topic mixture of topic models might eventually lead to different optimal choice of T (the number of topics) for different schemes. For the MSG scheme, we are modeling the number of topics existing in messages. Since a message is short and the number of messages is huge, we usually need a larger number of topics to obtain a reasonable model. On the other hand, for the USER scheme, we are modeling the number of topics for users. We can arguably say that each user may only have a relatively small number of topics that they are interested in and the total number of users are comparatively smaller than the volume of messages. Hence, through our experiments, the optimal number of topics is usually smaller than its in MSG scheme.

Note that in this paper we only explore schemes that do not require any significant modifications to the LDA or AT models. We do believe that better extensions of LDA which consider authors and messages simultaneously might be more useful.

4. EXPERIMENTS

In this section, we present the experimental evaluation of the schemes discussed in the previous section. For the experiments we use Twitter data obtained through both the streaming and normal APIs. We begin by describing some preprocessing steps of our data. Then, we test a variety of schemes discussed in the previous section on two realistic tasks. By studying the results, we will show that topic modeling is a powerful tool for short text messages.

4.1 Tasks

In our experiments, we have two different tasks, whose performance can be potentially enhanced by topic modeling techniques:

- Predicting popular Twitter messages

- Classifying Twitter users and corresponding messages into topical categories

For the first task, we consider the number of times a message has been retweeted as a measure of popularity. Therefore, we convert the problem into predicting whether a message will be retweeted in the **future**. Since we only have an incomplete set of Twitter messages and we cannot directly recover complete retweet patterns, we need to construct a reasonable dataset from our sample. Consider a collection of messages, some of which are duplicates of others. Before we measure if two messages are "similar", we take the following preprocessing steps: 1) We remove links from the messages; 2) We remove any word stating with the "@" character; 3) We remove non-latin characters in the message and convert all characters to lower case; and, 4) We calculate the hash value of all the messages. We use MD5 to obtain the signature for all messages. If two messages share the same MD5 value, we define them as "similar" to each other. We group similar messages together and sort them by time. All the versions of a message form a chain. For all messages in the chain except the first, we further filter out those messages without "RT". In other words, it does not matter if the first message is a retweet, but all subsequent messages in the chain must be retweets. For all filtered chains, if there are n messages in a particular chain, we take the first $n-1$ messages as "positive instances", which means they will be retweeted in the future, and the last one as "negative instance". In addition, all other messages which are not in any chains are also considered as "negative instances". Our task is to correctly predict all "positive instances" in the dataset.

The second task is more straightforward. In several Twitter directories (e.g., http://www.wefollow.com) and in the official Twitter site, lists of users with categories associated with them is provided. We take more than 250 verified users from the official Twitter Suggestions categories[1] under the assumption that these verified accounts are recognized as valid by real people and organizations. The categories do not overlap. We monitored the latest 150 messages generated by these users and try to classify the messages and the account into their corresponding categories which we obtained from Twitter Suggestion, under the assumption that these verified users strongly adhere to their corresponding categories that most of the messages generated by them are in the same topic.

Prior to attempting the two tasks, we also studied the topics learned by the models empirically mainly from two aspects: 1) Whether the topics obtained by different schemes are similar or not; and, 2) What is the quality of the topics. We compare the topics in both qualitative and quantitative ways.

4.2 Dataset

Our experiments employ the data from Twitter's APIs[2]. For the first task, we collected messages through Twitter's Streaming API, which is a push-style API with different levels of access which constantly delivers a small fraction of Twitter messages over a permanent TCP connection. We were granted the "Garden-hose" level of access at that time, which the company describes as providing a "statistically significant sample" of the messages that flow through their system. In our experiments, we use messages from the first and second week of November 2009 but we also find similar results by conducting the same experiments on other weeks. In order to reduce the dataset to a reasonable size that can be used to evaluate the techniques easily, we remove all non-latin characters from the messages. In addition, we also remove the users who only appear once in our dataset, with their corresponding messages. This results in a dataset of 1,992,758 messages and 514,130 users. In our experiments, we neither remove stop words nor perform stemming on the words. We replace all URLs with the word "link" and keep all hash tags. Therefore, we have 3,697,498 distinct terms for the two weeks of data.

[1]http://twitter.com/invitations/suggestions

[2]http://dev.twitter.com/

Table 1: Users From Twitter Suggestions

Category ID	Category Name	# of Users
0	Art & Design	3
1	Books	3
2	Business	8
3	Charity	15
4	Entertainment	42
5	Family	4
6	Fashion	5
7	Food & Drink	19
8	Funny	23
9	Health	9
10	Music	43
11	News	16
12	Politics	27
13	Science	4
14	Sports	39
15	Technology	22

For the second task, we crawled 274 verified users of 16 categories from Twitter Suggestion and their last 150 messages if available. In order to classify users, we aggregate all the messages generated by the same user into a giant document, denote as a "user profile". Similarly, we do not remove stop words and do not perform stemming. Thus, the dataset contains 52,606 distinct terms and 50,447 messages in total. The detailed number of users per category is shown in Table 1.

4.3 Evaluation Metrics & Parameters Setting

We cast both tasks into classification problems where the first one is to classify messages into retweets and non-retweets (note, "retweets" represent the messages will be retweeted in the **future**) and the second is to classify messages and users into topical categories. The baseline method for both tasks is a classifier using TF-IDF weighting values as the features.

For the first task, our basic evaluation scheme is to train the classifier on the first week and test it on the second week while for the second one, a simple cross-validation scheme is used. For the first task, we use Precision, Recall and F-Measure (F1 score) as the evaluation metric with their definitions shown as follows:

$$\text{Precision} = \frac{\text{number of true positives}}{\text{number of true positives + false positives}}$$

$$\text{Recall} = \frac{\text{number of true positives}}{\text{number of true positives + false negatives}}$$

$$\text{F-Measure} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

The vast majority of the instances in our dataset are negative ones (e.g., the messages will not be retweeted in the future). Therefore, a naive classifier may easily achieve more than 90% accuracy by choosing every instance as negative, which does not make much sense in our case. Hence, we do not report any results based on accuracy for the first task. For the second task, we use classification accuracy as the evaluation metric. We not only look at the classification accuracy for each category but also care about the overall classification accuracy.

Throughout the experiments, we use L2-regularized Logistic Re-

Table 2: "Similar" Topics Found by JS Divergence

The Topic Obtained by MSG scheme
[link] our from help world their people news more haiti red photo every two school end american change water million learn women through visit america fight money far girls national wine save young office children giving earth month community needs local trip relief future project malaria uk ones #haiti number program college south power donate launch between worth education full others students history safe room group lives summer during california earthquake past charity

The Topic Obtained by USER scheme
[link] rt and we day on your is us help haiti are by from you new world with about this have red people support at thanks join out will more great twitter can their up water read video w check today were make work here get photo what please last be women live kids an school children who save event vote now project relief pls malaria life #haiti friends every them has watch donate team thank follow sign global text keep working thx do need free learn earthquake many community million

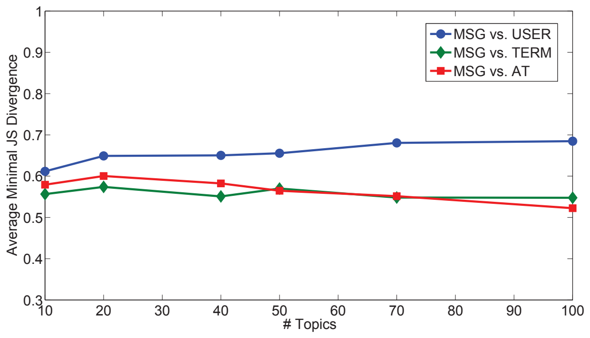

Figure 1: The Average Minimal JS Divergence

gression as our classifier[3]. In our preliminary experiments, we also tried L1 regularization, which corresponds to learning a sparse representation of features. Since we did not find any performance gains through L1 regularization, we only report the results on the L2 regularized classifier.

All the topic models used in the experiments have symmetric Dirichlet priors. We notice that asymmetric priors may lead to better results, suggested by [18]. However, in order to reduce the effect of optimizing hyper-parameters, we fix all of them to symmetric Dirichlet priors. More specifically, for β, we set it to 0.01 in all experiments and for α, we adopt the commonly used $50/T$ heuristics where T is the number of topics. In our experiments, we use Collapsed Gibbs Sampling [5] with speed-up techniques introduced in [20], which can be scaled to our large dataset.

4.4 Topic Modeling

In this section, we mainly study two questions: 1) whether different training schemes cause the model to learn different topics from the dataset; and, 2) what is the quality of topics learned from the dataset by different schemes. The dataset we used in this subsection is the topical classification dataset described in Section 4.1.

In order to answer the first question, we need to map topics learned by different schemes. Due to the "exchangeable" property of topic models [3], the topics learned from different runs of the models are not directly correspond, even for the exactly same settings. Therefore, a mapping process is required to find same or similar topics. In this work, we use Jensen-Shannon (JS) divergence to measure the similarity between topics. The JS divergence is a symmetric measure of the similarity of two pairs of distributions. The measure is 0 only for identical distributions and approaches infinity as the two differ more and more. Formally, it is defined as the average of the KL divergence of each distribution to the average of the two distributions:

$$D_{JS} = \frac{1}{2} D_{KL}(P||R) + \frac{1}{2} D_{KL}(Q||R)$$
$$R = \frac{1}{2}(P + Q)$$

where $D_{KL}(A||B)$ represents the KL divergence between variable A and B. In our case, the KL divergence is calculated as follows:

$$D_{KL}(A||B) = \sum_{n=1}^{M} \phi_{na} \log \frac{\phi_{na}}{\phi_{nb}}$$

where M is the number of distinct term types and ϕ_{na} is the probability of term n in topic a. For each topic i, we obtain a corresponding topic j with the minimal JS divergence score where topic i and j are trained through different schemes.

Let us first look at the results qualitatively. In Table 2, we list two topics identified by minimal JS divergence as "similar topics" where two models are trained on the dataset for the second task and the number of topics $T = 10$. The upper part of the table shows the topic found by the MSG scheme and the bottom part shows the topic obtained by the USER scheme. All the terms shown in the table are the topic terms sorted by ϕ scores. In other words, these terms are generated by the topics with high probabilities. Not very surprisingly, the top terms found by different schemes do not match with each other exactly. However, by carefully reviewing the terms, we find that most of them are related to some news events (e.g., Haiti earthquake) and politics.

In order to better quantify the difference between topics, we use two metrics based on JS divergence. One is to calculate the average divergence between "similar" topics, which we denote "the average minimal JS divergence". More specifically, for each topic i, we first find a "similar" topic j with minimal JS divergence. Then, we calculate the average of JS divergence over all discovered "similar" topics. Figure 1 displays the average minimal JS divergence between different models. In this figure, we see that there is obvious difference between topics learned by different schemes or models. Topics learned by the USER scheme are substantially different from the topics learned by the MSG scheme and JS divergence slightly increases with increasing number of topics. Compared to the USER scheme, topics learned by the TERM scheme and the AT

[3]http://www.csie.ntu.edu.tw/~cjlin/liblinear/

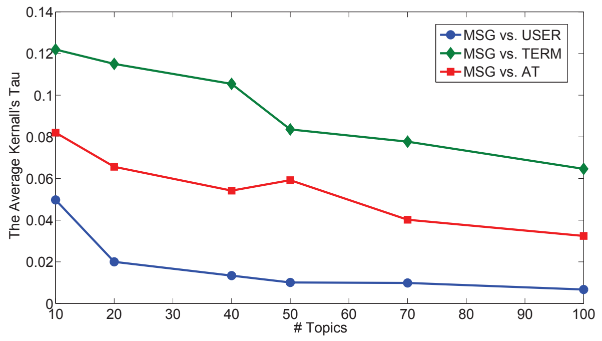

Figure 2: The Average Kendall's τ

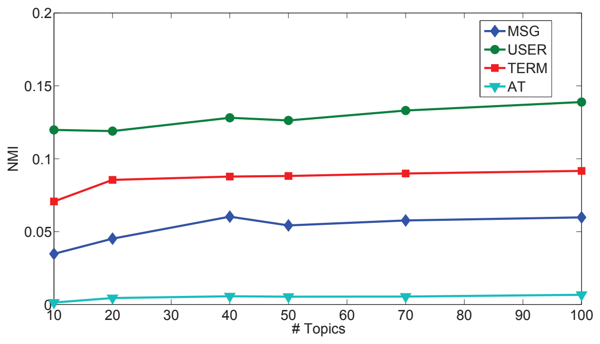

Figure 3: Normalized Mutual Information

model are closer to the topics of the MSG scheme. Note that almost all the JS divergence values are far from 0, which indicates that the probabilities of terms in each topic indeed differ apart.

From JS divergence, we conclude that the probabilities learned are different but we do not know how these difference may influence the relative positions of terms ranked in the topics. Therefore, the second metric we use is to measure the difference between rankings of terms obtained by topics. As shown in Table 2, while some of the terms found by different schemes are all ranked highly (e.g., haiti, relief), the exact ranking position is not the same. By looking at the discrepancy between rankings, we can understand how topics deviate from each other and how different models agree with each other. Here, we use Kendall's τ to measure the agreement between rankings. Given two different rankings of the same m items, Kendall's τ is defined as:

$$\tau = \frac{P - Q}{P + Q}$$

where P is the number of pairs of items in two rankings that are concordant and Q is the number of pairs of items in two rankings that are not concordant. τ ranges from -1 to 1, with 1 meaning the two rankings are identical and -1 meaning one is in the reverse order of the other. If $\tau = 0$, it means that 50% of the pairs are concordant while 50% of the pairs are discordant. We take the top 500 terms ranked by "similar" models identified by minimal JS divergence and calculate the τ values. Figure 2 shows the results of τ values between "similar" topics. Two immediate observations can be discovered. First, the disagreement between the MSG scheme and the USER scheme is substantially larger than other schemes. Second, as the number of topics increases, the disagreement increases.

Table 3: The Comparison of Performance on Retweet Prediction

Scheme	Precision	Recall	F1
TF-IDF	0.4216	0.3999	0.4105
MSG (100)	0.5088	0.2837	0.3643
USER (40)	0.6075	0.3677	0.4581
TERM (70)	0.5292	0.3061	0.3879
AT (70)	0.4811	0.2654	0.3421
TF-IDF + MSG	0.5150	0.3546	0.4200
TF-IDF + USER	0.6142	0.3897	0.4768
TF-IDF + TERM	0.5303	0.3582	0.4276
TF-IDF + AT	0.4736	0.3622	0.4104

Next, we would like to know the quality of topics found by the models. The dataset we used is still the topical classification dataset containing sixteen categories. Since we know the ground truth label of all the messages in the dataset (their categories), we can measure the quality by how likely the topics agree with the true category labels. Here, we use Normalized Mutual Information (NMI), which can be defined as follows:

$$\text{NMI}(\Omega, \mathbb{C}) = \frac{I(\Omega, \mathbb{C})}{[H(\Omega) + H(\mathbb{C})]/2}$$

where $I(\Omega, \mathbb{C})$ is mutual information between set Ω and $\mathbb{C}$ and $H(A)$ is the entropy. NMI is always a number between 0 and 1. NMI may achieve 1 if the clustering results can exactly match category labels while 0 if two sets are independent. Details of the calculation of NMI can be found in [9]. For each message, we use the maximum value in topic mixture θ to determine its cluster, which leads to a "hard" clustering result. After this mapping process, we compute NMI with the labels and the results are shown in Figure 3. From the figure, we see that NMI values are low in general. Clusters assigned by the USER scheme matches labels significantly better than other schemes. The NMI values by the AT model are nearly zero, indicating that they almost do not match class labels at all. As discussed before, the AT model does not provide a fully formalized generation process for documents. Therefore, the quality of topic mixture learned for messages is comparatively poor.

In conclusion, topics obtained by different schemes usually vary substantially. As shown in the experiments, the USER scheme might achieve better agreement with predefined labels, if available.

4.5 Predicting Popular Messages

In this section, we would like to see how the schemes and models discussed can influence classification performance. Here, we consider the problem of predicting potential "retweets". Remember, we treat the problem as a classification problem where the input is a set of features and the output tells us whether the target message will be retweeted in the future or not.

We first use TF-IDF weighting scores as features and train a Logistic Regression classifier. The result is shown in the first row of Table 3. Then, we train topic models according to the different schemes and obtain topic mixture θ for both messages and authors as introduced in the Section 3. For different schemes, we only report the best performance and its corresponding number of topics. We only test the number of topics in the range of 20 to 150. The results are shown from the second row to the fifth row (the first half of the Table) in Table 3. The first conclusion we can draw is that most of results are worse compared to the baseline, TF-IDF, while only the topics trained by USER scheme significantly outperform the baseline. In the last sub-section, we see that the topics trained

Table 4: The performance of TF-IDF features on Message Classification

Category	Accuracy	Category	Accuracy
0	0.3000	1	0.2143
2	0.2756	3	0.5909
4	0.4722	5	0.1250
6	0.2577	7	0.3553
8	0.3459	9	0.6471
10	0.5544	11	0.4026
12	0.5350	13	0.3553
14	0.6220	15	0.4185
		Average:	0.4792

Table 5: The best performance of USER Scheme on Message Classification

Category	Accuracy	Category	Accuracy
0	0.5000	1	0.0000
2	0.5128	3	0.9583
4	0.8223	5	0.0000
6	0.3814	7	0.8899
8	0.9082	9	0.7386
10	0.8718	11	0.8636
12	0.8132	13	0.5263
14	0.9330	15	0.9022
		Average:	0.8291

Table 6: The best performance of MSG Scheme on Message Classification

Category	Accuracy	Category	Accuracy
0	0.5000	1	0.3036
2	0.1218	3	0.9583
4	0.6934	5	0.0000
6	0.1753	7	0.8899
8	0.8894	9	0.8693
10	0.8277	11	0.7403
12	0.7749	13	0.5263
14	0.9732	15	0.8451
		Average:	0.7838

Table 7: The best performance of TF-IDF + USER on Message Classification

Category	Accuracy	Category	Accuracy
0	0.3000	1	0.2500
2	0.2692	3	0.5985
4	0.4776	5	0.1250
6	0.2680	7	0.3491
8	0.3388	9	0.6797
10	0.5492	11	0.4026
12	0.5478	13	0.3816
14	0.6327	15	0.4266
		Average:	0.4838

by USER scheme achieve higher NMI value, which implies that USER scheme might more likely match the underlying category information. Although other schemes do not perform well, we notice that the Precision is improved by all these schemes. If we argue that Precision is more valuable in this task (because once we make a "positive" decision, we have less chance to be wrong), we can conclude that topic models indeed help us.

Some literature [3] suggested that if we solely use topic mixture as features, we may not achieve better performance than TF-IDF. Thus, we combine topic model features and TF-IDF features and obtain the results in the second half (from 6th row to the bottom) of the Table 3. The results are trained on a classifier using the best performing topic model features with TF-IDF features. We can see that most of them improve performance and TF-IDF with USER scheme outperforms the previous best one that only uses the topic features. Surprisingly, the AT model performs the worse in the experiments and combining TF-IDF features does not give the AT model much boost in the performance.

In this task, we see that although sometimes topic features may not outperform simple TF-IDF features, it is good practice to combine them. USER scheme consistently provides good results, compared to other models.

4.6 User & Message Classification

In this section, we will see the results of the second task, classifying messages and authors into topical categories. First, let us turn our attention to the performance on message classification. Recall that we have 274 users from 16 categories in the dataset. For each user, we assume that all the messages generated by this user fall into the same category as the user. Therefore, for message classification, we use 90% of messages for training and 10% for testing and report the results on 5-fold cross validation. The baseline method is to use the TF-IDF weighting scores as features to train the classifier, which is shown in Table 4. Note that the category ids correspond to the categories introduced in Table 1. The overall accuracy is around 47% where the high performance is achieved in "Health" and "Sports" categories.

Again, similar to the first task, we use the topic mixture θ for both messages and users learned by topic models as features. We test the features in two settings, only using topic features and combining with TF-IDF features. We only report the best performance with its number of topics while we test the topic numbers from 10 to 150. Table 5 shows the best results obtained by USER scheme when the number of topics $T = 50$. Note, the overall accuracy is significantly improved and it is almost twice as accurate as raw TF-IDF features. However, we also note that the classifier results in zero accuracy in some categories. Category 1 ("Books") and category 5 ("Family") are two cases where the classifier does not achieve one valid instance. One potential reason for this phenomenon is that the number of instances in these categories are significantly smaller than other categories, which prevent the classifier and topic models to learn enough information about them. Table 6 shows the best results by the MSG scheme as $T = 100$. First, the overall accuracy is improved by TF-IDF features but lower than USER scheme. Second, we still have "Family" category with 0 accuracy. Due to space limits on the paper, we do not include detailed performance results for the TERM scheme and the AT model. The highest accuracy achieved by the TERM scheme is 0.6684 with $T = 100$ and by the AT model is 0.5459 when $T = 150$. Both of them are far worse than the MSG and USER schemes but still better than raw TF-IDF scores. When we combine topic features with TF-IDF features, unlike the first task shown in the last sub-section, the performance is always worse than only using topic features and only slightly better than solely using TF-IDF values. We only report the best results in Table 7, which is trained through USER scheme with $T = 40$. We notice that by combining TF-IDF features we can avoid the "zero" accuracy situation in all our experiments. Therefore, to some extent, TF-IDF features can capture some micro-level

Table 8: The Performance of TF-IDF on User Classification

Category	Accuracy	Category	Accuracy
0	0.5000	1	0.6667
2	1.0000	3	1.0000
4	0.9756	5	0.5000
6	0.4000	7	0.7895
8	0.8261	9	0.8750
10	0.9767	11	0.8750
12	1.0000	13	0.5000
14	0.9474	15	0.8636
		Average:	0.9051

Table 9: The Best Performance of USER on User Classification

Category	Accuracy	Category	Accuracy
0	0.0000	1	0.0000
2	0.0000	3	0.5333
4	0.5610	5	0.0000
6	0.0000	7	0.1053
8	0.0000	9	0.5000
10	0.6279	11	0.0000
12	0.7600	13	0.0000
14	0.7895	15	0.3182
		Average:	0.4380

characteristics of categories while the topic features are usually too high level (since the feature is indeed topic mixture not the topic distribution itself).

Now, let us turn to the problem of classifying users into topical categories. Similar as message classification, we split 90% of messages and aggregate the messages in training set for each user to build the user profiles. So, the training user profiles and testing profiles are always different and do not mixed. Again, TF-IDF is calculated as features for user profiles, which are aggregations of all messages generated by the same user. The baseline is shown in Table 8. Surprisingly, the performance is very high, almost twice higher than the baseline in message classification. For category "Business" and "Charity", the classifier distinguished all instances successfully. In fact, in our experiments, the classifier trained on topic features performs much worse than the baseline regardless of schemes. We only report the best performing results in Table 9, which is obtained through USER scheme with $T = 20$. We notice that not only the overall accuracy is not as good as TF-IDF features but using topic features also results in several zero accuracy in different categories. One reason is again the content in those categories is limited. An interesting point is that if we combine TF-IDF features with topic features, the overall performance is still around 90% (in fact, only with marginal improvement). Remember, for user profiles, we crawled the latest 150 updates for each user, if available. Therefore, for most users, the profile already contain enough information to learn. This situation is significantly different from message classification where we have the problem of sparsity.

Compared to the results on message classification where topic features play an important role to improve the performance and user classification where topic features fail to outperform the baseline, we believe that topic models can help us model short text while for longer content, more sophisticated models might be required to improve performance (e.g., Supervised LDA [2], Label LDA [14]).

5. DISCUSSION & SUMMARY

Although we do not introduce new topic models to address the issues of short text modeling especially in microblogging environments in this paper, our work sheds some light on how research on topic models can be conducted for short text scenarios. More specifically, through our experiments, we demonstrate that the effectiveness of trained topic models can be highly influenced by the length of the "documents"; namely, a better model can be trained by aggregating short messages. This argument has attracted little attention in the research community in the past and should be justified through more thorough experiments and theoretical analysis. In addition, our empirical study demonstrated that topic modeling approaches can be very useful for short text either as solely used features or as complementary features for multiple real-world tasks. (Note that this does not mean that the model itself should be trained on short text and we show that a model trained on aggregated longer text can yield better performance.) We also showed that when content information is already large enough (e.g., in user classification), topic models become less effective compared to simple TF-IDF scores. Moreover, through the experiments, we showed that the simple extension to the AT model does not yield better modeling for messages and users and indeed it is worse than training a standard LDA model on user aggregated profiles. We conjecture that the reason may be the "OR" nature of the AT model while a message is either "generated" by the message or by an author. We suggest that future models might examine how to model a hierarchical structure between users and messages.

In this paper, we conducted extensive qualitative and quantitative experiments on three proposed schemes based on standard LDA and one extended model based on the AT model. We compared a number of aspects of these schemes and models, including how the topics learned by these models differ from each other and their quality. In addition, we showed how topic models can help other applications, such as classification problems. In the experiments we demonstrated that topic models learned from aggregated messages by the same user may lead to superior performance in classification problems and topic model features can improve performance in general, especially when the research targets are messages.

Acknowledgments

This work was supported in part by a grant from the National Science Foundation under award IIS-0545875 and an equipment grant from Sun Microsystems. We also thank Ovidiu Dan for collecting the Twitter data and for many helpful discussions.

6. REFERENCES

[1] D. Blei and J. Lafferty. Topic models. *Text Mining: Theory and Applications*, 2009.

[2] D. M. Blei and J. D. Mcauliffe. Supervised topic models. *Advances in Neural Information Processing Systems 21*, 2007.

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

[4] J. Chang, J. Boyd-Graber, and D. M. Blei. Connections between the lines: augmenting social networks with text. In *KDD '09: Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 169–178, 2009.

[5] T. L. Griffiths and M. Steyvers. Finding scientific topics. *Proceedings of the National Academy of Sciences of the United States of America*, 101:5228–5235, 2004.

[6] A. Java, X. Song, T. Finin, and B. Tseng. Why we twitter: understanding microblogging usage and communities. In *WebKDD/SNA-KDD '07: Proceedings of the 9th WebKDD and 1st SNA-KDD Workshop on Web mining and Social Network Analysis*, pages 56–65, 2007.

[7] B. Krishnamurthy, P. Gill, and M. Arlitt. A few chirps about twitter. In *WOSP '08: Proceedings of the First Workshop on Online Social Networks*, pages 19–24, 2008.

[8] Y. Liu, A. Niculescu-Mizil, and W. Gryc. Topic-link lda: joint models of topic and author community. In *ICML '09: Proceedings of the 26th Annual International Conference on Machine Learning*, pages 665–672. ACM, 2009.

[9] C. D. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.

[10] A. McCallum, X. Wang, and N. Mohanty. Joint group and topic discovery from relations and text. In *Statistical Network Analysis: Models, Issues and New Directions*, volume 4503 of *Lecture Notes in Computer Science*, pages 28–44. Springer-Verlag, Berlin, Heidelberg, 2007.

[11] R. M. Nallapati, A. Ahmed, E. P. Xing, and W. W. Cohen. Joint latent topic models for text and citations. In *KDD '08: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 542–550, 2008.

[12] X.-H. Phan, L.-M. Nguyen, and S. Horiguchi. Learning to classify short and sparse text & web with hidden topics from large-scale data collections. In *WWW '08: Proceedings of the 17th International Conference on World Wide Web*, pages 91–100, 2008.

[13] D. Ramage, S. Dumais, and D. Liebling. Characterizing microblogs with topic models. In *International AAAI Conference on Weblogs and Social Media*, 2010.

[14] D. Ramage, D. Hall, R. Nallapati, and C. D. Manning. Labeled lda: a supervised topic model for credit attribution in multi-labeled corpora. In *EMNLP '09: Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 248–256. Association for Computational Linguistics, 2009.

[15] M. Rosen-Zvi, C. Chemudugunta, T. Griffiths, P. Smyth, and M. Steyvers. Learning author-topic models from text corpora. *ACM Transactions on Information Systems*, 28(1):1–38, 2010.

[16] M. Rosen-Zvi, T. Griffiths, M. Steyvers, and P. Smyth. The author-topic model for authors and documents. In *UAI '04: Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence*, pages 487–494, 2004.

[17] M. Sahami and T. D. Heilman. A web-based kernel function for measuring the similarity of short text snippets. In *WWW '06: Proceedings of the 15th International Conference on World Wide Web*, pages 377–386, 2006.

[18] H. M. Wallach, D. Mimno, and A. McCallum. Rethinking lda: Why priors matter. In *Proceedings of the 23rd Annual Conference on Neural Information Processing Systems*, 2009.

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

[20] L. Yao, D. Mimno, and A. McCallum. Efficient methods for topic model inference on streaming document collections. In *KDD '09: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 937–946, 2009.

[21] W.-T. Yih and C. Meek. Improving similarity measures for short segments of text. In *AAAI'07: Proceedings of the 22nd National Conference on Artificial Intelligence*, pages 1489–1494, 2007.

[22] H. Zhang, C. L. Giles, H. C. Foley, and J. Yen. Probabilistic community discovery using hierarchical latent gaussian mixture model. In *AAAI'07: Proceedings of the 22nd National Conference on Artificial Intelligence*, pages 663–668, 2007.