Topic Modeling in Twitter: Aggregating Tweets by Conversations

David Alvarez-Melis*, **Martin Saveski***
Massachusetts Institute of Technology
Cambridge, MA, USA
{dalvmel, msaveski}@mit.edu

Abstract

We propose a new pooling technique for topic modeling in Twitter, which groups together tweets occurring in the same user-to-user conversation. Under this scheme, tweets and their replies are aggregated into a single document and the users who posted them are considered co-authors. To compare this new scheme against existing ones, we train topic models using Latent Dirichlet Allocation (LDA) and the Author-Topic Model (ATM) on datasets consisting of tweets pooled according to the different methods. Using the underlying categories of the tweets in this dataset as a noisy ground truth, we show that this new technique outperforms other pooling methods in terms of clustering quality and document retrieval.

Introduction

The problem of characterizing text documents based on their content is one of great importance in machine learning and natural language processing. Although sometimes a goal by itself, this characterization is often used as an input for tasks that involve retrieving, classifying or even predicting documents. In the context of social media, studying the characteristics of text in posts and messages is crucial for many tasks such as breaking news detection, friend recommendation and sentiment analysis, among others.

When documents lack annotations or category labels—as is often the case in practice, particularly for microblogging feeds—one must rely on unsupervised approaches that discover underlying topics directly from the raw text features in the documents, such as Latent Dirichlet Allocation (LDA) (Blei, Ng, and Jordan 2003) or the Author Topic Model (ATM) (Rosen-Zvi et al. 2004). However, these methods are designed to be used on documents that are sufficiently long to extract robust per-document statistics. When applied directly on posts from microblogging platforms, which are usually short and often noisy, these methods result in topics that are uninformative and hard to interpret.

An intuitive solution is aggregating related tweets (*pooling*) and applying standard topic modeling techniques on these pooled documents. This approach is appealing since it allows for the use of optimized off-the-shelf topic modeling toolkits. Two of the most popular pooling techniques involve merging into a single document all the tweets posted by a user (Hong and Davison 2010) or those containing the same hashtag (Mehrotra et al. 2013). These approaches tend to improve upon running topic modeling on unpooled tweets, yet often result in documents that are not topically homogeneous, because their underlying assumptions about topic consistency within users and hashtags are frequently violated. In addition, pooling by hashtags results in considerable duplication of tweets and thus longer training times.

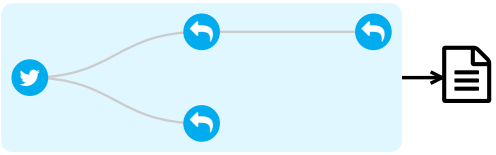

Figure 1: Pooling by conversations. A document consists of a seed tweet, the tweets written in reply to it, tweets written in reply to these and so on.

In this study, we propose a new scheme for pooling tweets into longer documents based on conversations. In this framework, a document consists of a seed tweet, all the tweets written in reply to it posted by other users and the replies of the original poster to these (Figure 1). The motivation behind this technique is that user-to-user interaction in Twitter tends to revolve around a few related topics, so pooling tweets by conversations can lead to a more coherent document aggregation and hence more relevant topic extraction.

We evaluate this novel pooling technique by means of a comprehensive empirical comparison against state-of-the-art pooling techniques. For this purpose, we train both LDA and ATM topic models and we construct a dataset in a way that provides implicit topical labels for each tweet, allowing us to assess the quality of the models. We evaluate the various pooling techniques in terms of clustering quality and their performance on a document retrieval task.

The experimental results show that this new technique performs favorably against alternative pooling schemes. In particular, our method yields topics that perform better than those derived by pooling by hashtags, but takes considerably less time to train.

*Both authors contributed equally to this work.

Tweet-pooling Schemes

Topic modeling techniques such as LDA and ATM have been most successfully applied to corpora composed of long documents with regular vocabulary and grammatical structure, such as news and scientific articles. Tweets, on the other hand, are: (i) short, less than 140 characters; (ii) mixed with contextual clues, such as URLs, hashtags, or Twitter names; and (iii) use informal language with misspellings, acronyms, emoticons, and nonstandard abbreviations. While applying linguistic preprocessing may somewhat help (Han, Cook, and Baldwin 2012) (addressing problems (ii) and (iii)), the essential problem, the size of the documents, remains. Tweets, when considered as individual documents, are too short to produce robust per-document term co-occurrence statistics. Indeed, previous studies show that standard topic modeling techniques fail to identify the essential information in tweet collections, discovering topics that are noisy and hard to interpret (Zhao et al. 2011).

An intuitive solution to address the issue of short documents in Twitter is *tweet pooling*: merging related tweets together and applying standard topic modeling techniques on the resulting pooled documents. The goal of a pooling scheme is to aggregate tweets that are topically similar, obtaining documents that are not only longer but are also topically coherent. This allows for the discovery of better topics without modifying the existing topic modeling techniques.

Drawing upon existing work we consider two different pooling schemes based on users (Hong and Davison 2010) and hashtags (Mehrotra et al. 2013). Other possible pooling techniques have been proposed in the literature, but the empirical evidence of their advantage over unpooled tweets is less compelling (Mehrotra et al. 2013), so we do not consider them in this study. These two standard pooling techniques and the conversation-based scheme proposed here are explained below in detail.

Tweet-pooling (Unpooled). The default approach that treats each tweet as a single document. This serves as our baseline for comparison to pooled schemes.

User-pooling. In this scheme a document consists of all tweets posted by a single user, so there are as many documents as users. This popular technique has been shown to be superior to tweet-pooling (Hong and Davison 2010).

Hashtag-pooling. A document consists of all tweets that contain a given hashtag. There are as many documents as hashtags, but since tweets may have multiple hashtags, a tweet may appear in several documents. Tweets that do not contain any hashtags are considered as individual documents, i.e., as if they contain a hashtag that does not appear in any other tweet. There is some empirical evidence that this pooling scheme yields better topics than pooling by users (Mehrotra et al. 2013), but training times tend to be much longer.

Conversation-pooling. A document consists of a tweet, all the tweets written in reply to it, the replies of the replies, and so on (Figure 1). Since a tweet may have multiple replies and each of those may in turn have many replies, the resulting *conversation tree* can span several tweets and authors.

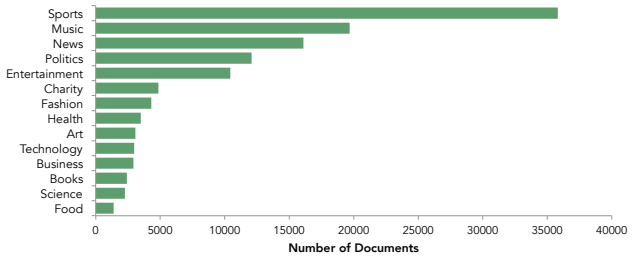

Figure 2: Distribution of latent categories in the dataset.

Tweets that are not replies or do not have any replies (single node trees) are considered as individual documents. Since these conversation trees tend to be on the same (or at least related) topics, we expect the documents constructed in this way to be more homogeneous than those obtained with previous approaches.

Experimental Setup

Data collection. We construct a dataset that attempts to capture a wide range of topics. Similar to (Hong and Davison 2010), we start with a set of 14 topics (Figure 2) and for each topic we select the top 25 most influential users. To identify them we use the website `http://www.wefollow.com`, which ranks users per topic according to a pagerank-like score. Next, we use the GNIP Historical API to retrieve all public tweets posted by these users as well as all tweets that mention them during a period of one week in April 2014. We chose this strategy because of two reasons: (i) Querying for user mentions allows us to recover most of the conversations in which these users participated, although some conversation trees may be broken down if the replies do not mention the seed users (e.g., if there is a conversation tree $t_1 \leftarrow t_2 \leftarrow t_3$, and t_2 is not in the database, we have no way of connecting t_1 and t_3). (ii) Retrieving tweets posted by users who are influential in certain topics provides a category for the tweets. Even though not all the tweets written by the seed authors will be on their associated topic, these latent categories give us a rough notion of the true topic and allows us to evaluate and compare the different pooling techniques and topic models. Using this strategy we retrieved 161,607 tweets. To build conversation trees (as in Figure 1) and aggregate tweets by conversations, we use the `in_reply_to_status_id` field returned by the API.

Preprocessing. We preprocess the tweets by lower-casing and removing all URLs, mentions (tags of other Twitter users) and stop-words. We also remove all tokens that consist only of non-alphanumeric characters (this includes all emoticons) and all short tokens ($<$3 characters). We then filter out all non-English and very short tweets ($<$3 tokens). We also remove tweets by extremely active users ($>$200 tweets/week) and very long conversations ($>$200 tweets) to avoid excessively long documents. Finally, we filter out all re-tweets (i.e., reposting tweets posted by other users), since they serve more as an endorsement rather than an original content. After these preprocessing steps, we end up with 101,522 tweets.

	Pooling Technique	Documents	Authors	Tokens
Train	Conversations	61,795	59,596	510,605
	Hashtags	64,789	59,596	643,746
	Users	59,596	59,596	510,605
	Tweets (not pooled)	81,218	59,596	510,605
Test	Tweets (not pooled)	20,304	12,371	120,751

Table 1: Dataset statistics.

Data Splits. We randomly split the tweet corpus into training (80%) and test sets (20%), preserving the same distribution of tweet categories in both splits. Due to the long training times of some models we were unable to run cross-validation (see Section "Run Times"). To be able to compute the topics of unseen documents using ATM we ensure that all users in the test set also appear in the training set. We then aggregate the tweets in the training set using the four pooling schemes, resulting in four different training sets.

Data Statistics. Table 1 shows various statistics of the training sets obtained by applying the different pooling schemes, and of the (unpooled) test set. Since tweets often contain multiple hashtags, pooling by hashtags results in a dataset containing 26% more tokens than the other schemes. Figure 2 shows the distribution of categories in the tweets. Although we have considered the same number of users for all categories (25), some categories are more represented than others, indicating that users associated with these categories tend to tweet more.

Training Configuration. We ran ATM and LDA using a popular topic modeling toolbox[1], which uses Markov Chain Monte Carlo sampling for inference. As suggested in (Rosen-Zvi et al. 2004), we set the Dirichlet priors to $\alpha = 50/T$ and $\beta = 0.01$. For each model, we ran 10 independent chains, using a burn-in time of 2,000 iterations, and then took samples every 100 iterations, until we collected 10 samples for each chain. When computing word probabilities per topic, we average the parameters learned across chains.

Results

Evaluating unsupervised topic models is usually challenging. In our setup, however, we can do so by exploiting the particular way in which the data was collected. Since our corpus was retrieved by querying for conversations involving a set of seed authors, each of which is associated with one of 14 categories, we have a raw notion of topic label for each document. We leverage these noisy labels to evaluate the topics produced by each model from two perspectives: clustering quality and document retrieval. Finally, we compare their efficiency in terms of running times.

Clustering Evaluation

In our first experiment, we cluster the test tweets by assigning the most likely topic predicted by our models, and compare these clusters with those implied by the underlying categories. To avoid having to directly map topics to categories, we use instead two standard clustering measures that quantify clustering quality against reference classes, *Normalized Mutual Information* (NMI) and *Adjusted Rand Index* (ARI) (Manning et al. 2008).

		(Scale: $\times 10^2$)	T = 20	T = 50	T = 100	T = 200	T = 300	T = 400
NMI	ATM	Conversations	**7.819***	7.420	**8.021**	**8.502***	**9.489***	**9.967***
		Hashtags	6.668	**7.810***	7.850	8.230	8.846	9.835
		Users	7.500	6.812	7.968	7.769	9.200	9.842
		Tweets	7.031	6.819	7.456	8.445	9.213	9.595
	LDA	Conversations	4.998	5.937	6.532	7.276	8.066	8.837
		Hashtags	6.559	5.995	7.011	7.532	8.595	9.270
		Users	7.537	6.983	6.933	7.734	8.399	9.313
		Tweets	4.562	5.363	5.946	6.636	7.513	8.201
Adj. Rand Index	ATM	Conversations	**4.514**	1.828	1.170	**0.654***	**0.542***	0.390
		Hashtags	2.875	**1.965***	1.173	0.583	0.406	0.376
		Users	4.407	1.702	**1.195**	0.517	0.478	**0.399***
		Tweets	3.637	1.549	0.973	0.628	0.464	0.351
	LDA	Conversations	2.551	1.469	0.867	0.515	0.332	0.297
		Hashtags	3.361	1.705	1.035	0.632	0.471	0.373
		Users	4.508	1.614	0.879	0.522	0.374	0.332
		Tweets	2.374	1.339	0.761	0.441	0.297	0.254

Table 2: Clustering metrics for various model/topic configurations. The numbers correspond to the means ($\times 10^2$) over 10 independent samples. The best result for each value of T is highlighted in bold. Asterisks denote values where the difference between the first and second best configurations is statistically significant ($p < 0.05$) using a two-sample t-test.

Table 2 shows ARI and NMI scores for several topic model sizes. For each model/topic configuration we present the mean computed over ten samples of the Monte Carlo chain. As expected, NMI increases with the number of topics, since this amounts to creating smaller clusters which are more likely to be homogeneous. The ARI, on the other hand, decreases as *T* increases since it penalizes for the discrepancy between the size of the clusters and the reference categories. In general, ATM models outperform their LDA counterparts in both metrics, with pooling by conversations and hashtags frequently achieving the best or second best result. ATM-Conversations has a clear advantage over ATM-Hashtags for $T \geq 100$.

Document Retrieval Evaluation

Next, we evaluate the topics discovered by the different pooling techniques on a document retrieval task. We treat every tweet in the test set as a query and return tweets from the training set according to their topic similarity to the query. Retrieved tweets are considered relevant if they have the same underlying category as the query tweet.

We proceed as follows. We use the topics learnt by the different pooling schemes to infer the topics of all train and test tweets[2]. For every test tweet (i.e., query tweet), we compute the cosine similarity between its topics and the topics of all train tweets, and we retrieve the top 10 most similar train tweets. Finally, we compare whether the categories of the retrieved tweets match the category of the test tweet.

[1] `http://psiexp.ss.uci.edu/research/programs_data/toolbox.htm`

[2] Note that for all pooling schemes except for tweets, the models are tested on documents in a different format (individual tweets, i.e. not pooled) from what they were trained on (pooled).

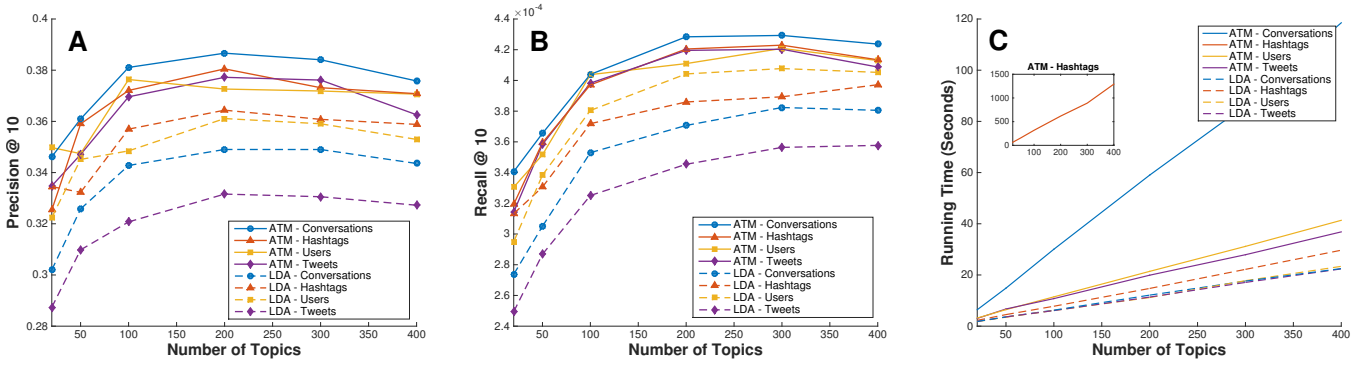

Figure 3: **A-B**: Evaluation of the topics discovered by the different pooling techniques on a document retrieval task. We measure precision (A) and recall (B) at 10. Pooling by conversations combined with ATM outperforms all other methods. **C:** Run times for 10 Gibbs-sampling iterations. ATM models take longer to train, with ATM-Hashtags requiring an order of magnitude longer sampling times than all other models. Figures best seen in color.

Figures 3A and 3B show Precision and Recall at 10 for all model configurations on this retrieval task. Note that this evaluation procedure accounts for the soft membership of documents to topics, as opposed to the clustering evaluation in the previous section where every document was assigned to the most likely topic. ATM performs better than LDA across pooling techniques in terms of both precision and recall. When running ATM, pooling by conversations outperforms all other pooling techniques, which have roughly the same performance. When running LDA, the user and hashtag pooling schemes perform best, followed by conversations and tweets. Overall, ATM-Conversations leads to the best performance.

Run Times

Finally, we compare the models in terms of the running time of their Gibbs-sampling sub-routine (Figure 3C). As expected, run times grow with the number of topics and training ATM models is generally slower than training their LDA counterparts, due to the additional inference over authors across documents. For LDA, the differences in training times between pooling techniques can be explained by the number of tokens in the corpus. Pooling by hashtags causes duplication of tweets and thus longer training times. For ATM, on the other hand, there is the additional impact of the number of authors per document. Pooling by conversations and by hashtags yields an average of 1.29 and 1.34 authors per document, respectively, while pooling by tweets or users results in only one author per document. Note that training ATM and pooling by hashtags is one order of magnitude slower than for any of the other pooling techniques.

Discussion

The experimental results shown here illustrate the advantage of aggregating tweets into longer documents as a preprocessing step for topic modeling. However, they also highlight two key challenges faced by these aggregation schemes: (i) modeling authorship information and (ii) producing topically coherent pooled documents.

The first of these challenges arises from the fact that pooled documents are very often composed of text written by multiple users. Using ATM to model this variation almost always leads to improvements over LDA (which ignores authorship information) across all pooling schemes, both on clustering (Table 2) and document retrieval (Figures 3A, 3B). An exception is pooling by users, which produces single-author documents and therefore does not benefit significantly from the additional flexibility of ATM.

The pooling method proposed here addresses challenge (ii) by leveraging the intuition that user-to-user conversations tend to be topically homogeneous. The empirical results support this hypothesis: overall, pooling by conversations combined with ATM outperforms all other configurations in both clustering (Table 2) and document retrieval (Figures 3A, 3B). This method and the best alternative, pooling by hashtags, perform comparably on clustering, but pooling by conversations yields considerably better results on document retrieval. Besides having better performance, the proposed method results in considerably lower training times than pooling by hashtags (Figure 3C).

References

Blei, D. M.; Ng, A. Y.; and Jordan, M. I. 2003. Latent dirichlet allocation. *Journal of Machine Learning Research*.

Han, B.; Cook, P.; and Baldwin, T. 2012. Automatically constructing a normalisation dictionary for microblogs. In *EMNLP*.

Hong, L., and Davison, B. D. 2010. Empirical study of topic modeling in twitter. In *Workshop on Social Media Analytics*.

Manning, C. D.; Raghavan, P.; Schütze, H.; et al. 2008. *Introduction to information retrieval*. Cambridge Press.

Mehrotra, R.; Sanner, S.; Buntine, W.; and Xie, L. 2013. Improving lda topic models for microblogs via tweet pooling and automatic labeling. In *SIGIR*.

Rosen-Zvi, M.; Griffiths, T.; Steyvers, M.; and Smyth, P. 2004. The author-topic model for authors and documents. *UAI*.

Zhao, W. X.; Jiang, J.; Weng, J.; He, J.; Lim, E.-P.; Yan, H.; and Li, X. 2011. Comparing twitter and traditional media using topic models. In *Advances in Information Retrieval*.