Finding Influencers within Fuzzy Topics on Twitter

Tal Stramer, Stanford University

1. Introduction

In recent years, there has been an explosion in usage of online social networks such as Twitter. This has led to an abundance of content around a wide variety of topics, which can often be overwhelming for users trying to find content around a specific set of interests. Previous work on topic-modeling Twitter content has largely focused on classifying tweets or authors into a static set of categories, like ‘Sports’ or ‘Politics’. In reality, however, interests tend to be more fine-grained and dynamic. For example, a user may be interested in the 2016 presidential election, the world series, or a recent natural disaster. Furthermore, a user is not always interested in consuming a firehose of content around a topic, but rather a filtered stream of content from just influential users for that topic.

The goal of this paper is two-folds: 1) to build a model to identify a fuzzy topic taxonomy on Twitter, and 2) to identify influential users within each topic. The word fuzzy is used to mean that the topic taxonomy is not manually defined or static, but rather is determined algorithmically based on what people are talking about on Twitter, and thus changes continuously. These topics are either very specific, like a tv show or video game, or time sensitive, like a trend, event, or natural disaster. A user is ‘influential’ for a given topic if they are likely to produce engaging content about that topic. Engaging content is identified by the number of likes, retweets, and replies a user gets on Tweets they post about the topic. The input to our algorithm is the top 200 search queries submitted on Twitter between 11/1/2015 and 11/7/2015 and all tweets (with engagement counts) by verified users between these dates that contain at least one of the top search queries. A variant of LDA is used to group search queries based on similarity in search queries contained in user’s tweets. Next, a mixture of Gaussian model is used to model how influential a user is for a topic based on tweet engagement counts and the LDA model. Finally, a multi-class SVM is used to predict how influential newly posted tweets will be for a given topic based on historical engagement counts of tweets by the same author about the topic.

2. Related Work

Previous work on topic-modeling Twitter content has largely focused on classifying tweets or authors into a static set of categories. Hong et al. [1] used LDA and a variant of the Topic-Author model to classify users into a set of 16 static categories, such as ‘Technology’ and ‘Sports’. Ramage et al. [2] used a variant of LDA to discover 4 different latent categories for tweets: events and ideas, social topic, personal updates, and trends in language usage. Both of these models consider all non-stopwords in a tweet, which leads to a lot of noise when determining the most relevant entities for a topic. Additionally, both of these models fail to take into account the time-sensitive nature of content on Twitter.

Vosecky et al. [3] tackled both of the above mentioned problems with a new model called the Multi-Faceted Topic Model (MfTM). Their model is a variant of LDA that allows for each latent topic to have a set of orthogonal ‘facets’. In their paper, they consider the following ‘facets’: general terms, person names, organizations, location names, and a temporal distribution. While their model produces a more fine-grained topic taxonomy, it does not consider signal for determining the popularity of a topic or which users are influential for a given topic.

Weng et al. [4] used a variant of PageRank to determine topic-specific influencers based on the follow graph. While this is a novel idea for finding influential users, the latent topics produced by their model contain a lot of noise due to taking into account all non-stopwords in tweets.

3. Dataset and Features

A dataset is built from U.S. Twitter activity between 11/1/15 and 11/7/15 based on internal Twitter APIs. A one-week period is chosen to be short enough that it is likely that each search query non-ambiguously refers to a single topic or event, but long enough that it will encompass topics or events spanning multiple days.

All search queries submitted by users between 11/1 and 11/7 are collected and sorted by frequency. The top 200 search queries represent the base entities for the topic taxonomy.

Search Query	Search Count
#aldubthevisitor	65,977
#aldubnewcharacter	58,081
greg hardy	53,872
#manonthemoon	50,721
ben carson	49,487

Figure 1: top 5 search queries submitted

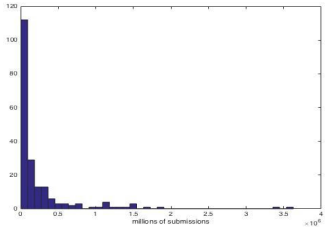

Figure 2: histogram of search query submission counts

All tweets by verified users between 11/1 and 11/7 that contain any of the top 200 search queries are collected. Tweets are grouped by user to obtain for each user the number of each search query contained in their tweets. Users having less than 5 instances of a top search query in their tweets are removed. After filtering, the final dataset contains 218,884 tweets and 4,930 users. Additionally, for each tweet, the number of favorites, retweets, and replies is collected.

Search Query	Tweet Count
christmas	3,627,502
halloween	3,391,114
exo	1,843,648
drake	1,652,856
trump	1,539,416

Figure 3: top 5 search queries by tweet count

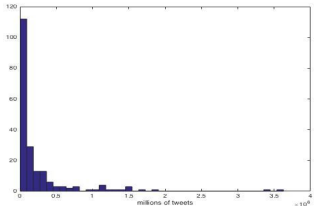

Figure 4: histogram of search queries tweet counts

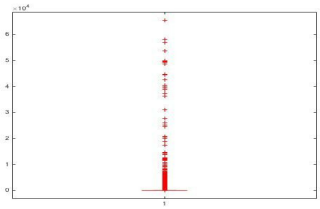

Figure 5: boxplot of tweet engagements

4. Methods

In order to extract a set of topics from the top search queries, a variant of Latent Dirichlet Allocation [5] (LDA) is used. LDA is a un-supervised learning algorithm that explains structure in a set of documents by representing each document as a probability distribution over a set of unobserved topics, and each topic as a probability distribution over a set of words. Specifically, LDA uses two probability distributions. The first is the Multinomial distribution, which models exactly k possible events and n trials. x_i is a variable representing the number of trials where event i occurs and p_i is the probability of event i occurring. The Multinomial distribution has the following probability mass function: $f(x_1, ..., x_k; n, p_1, ..., p_k) = \frac{n!}{x_1!...x_k!} p_1^{x_1} ... p_k^{x_k}$. The second is the Dirichlet distribution, which is a probability distribution over the

space of Multinomial distributions. To generate data , one first draws $\vec{p} \sim Dir(\alpha_1, ..., \alpha_k)$ and then draws $\vec{x} \sim Mult(\vec{p})$. The Dirichlet distribution has the following probability mass function: $f(p_1, ..., p_k; \alpha_1, ..., \alpha_k) = \frac{\prod_{i=1}^{k} \Gamma(\alpha_i)}{\Gamma(\sum_{i=1}^{k} \alpha_i)} \prod_{i=1}^{k} p_i^{\alpha_i - 1}$. The generative process of LDA is pictured in figure 6 below.

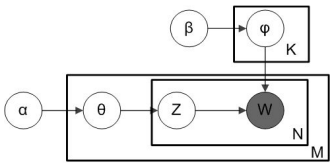

Figure 6 (left): α is a parameter of a Dirichlet distribution generating θ, the parameter of a Multinomial distribution generating topics for a document. β is a parameter of a Dirichlet distribution generating φ, the parameter of a Multinomial distribution generating words for a topic. Z is the topic assigned to word W.

The joint distribution of LDA for k topics, m documents, and n words is given by: $P(\theta, \varphi, Z, W|\alpha, \beta) = \prod_{j=1}^{k} P(\varphi_j|\beta) \prod_{i=1}^{m} P(\theta_j|\alpha) \prod_{i=1}^{m} P(z_{i,n}|\theta_i) P(w_{i,n}|z_{i,n}, \varphi_{z_{i,n}})$. The goal of LDA is to find values of the unknown parameters θ, φ, and Z that maximize this joint distribution. There is no explicit formula to solve this optimization problem, so Gibbs sampling is used instead, which is an algorithm that allows the estimation of a probability distribution P(x) through repeated sampling of $P(x_i|x_{i-1})$. The topic assignments are initialized randomly and then refined by repeatedly sampling topic assignments for each word until convergence. The parameters θ and φ are then computed from the topic assignments.

Tweets from each user are aggregated into a single 'document'. The 'document' for a user is represented as a bag-of-words, where each instance of a top search query appearing in a tweet by that user is considered a separate 'word'. One immediate problem is that search queries sometimes have ambiguous topic assignments. For example, the search query "Christmas" is referred to in the context of many different topics or events. These search queries are too ambiguous to be good representations for a single topic. In order to remove these search queries, successive iterations of LDA are run; after each iteration a search query that is among the top n most likely words in at least m topics is removed (specific values of m and n are discussed in the 'Results' section). The algorithm terminates after the first iteration in which no search queries are removed.

Based on the latent topics, the next step is to determine the most influential users for each topic. To that end, a model is built to determine how influential a tweet is for a specific top search query contained in the tweet based on the number of engagements the tweet receives relative to other tweets containing that search query. 'Influential' is modeled as a discrete variable taking on values between 1 and k. For example, for k=4, the buckets can roughly be thought of as 'no influence', 'low influence', 'medium influence', and 'high influence'. The majority of engagements for a specific search query follow a generalized pareto distribution. However, engagements for a specific search query follow an approximate normal distribution when conditioned on an 'influencer' level. Formally, given total engagements $e^{(i)}$ and an influencer level $z^{(i)}$ for a specific tweet and search query, the following modeling assumptions are made: $z^{(i)} \sim Multinomial(\varphi)$ and $e^{(i)}|z^{(i)} = j \sim N(\mu_j, \Sigma_j)$ (justified in the results section). The parameters φ, μ, and Σ are estimated for each search query by maximizing the likelihood function: $L(\varphi, \mu, \Sigma) = \prod_{i=1}^{m} p(x^{(i)}; \varphi, \mu, \Sigma) = \prod_{i=1}^{m} \sum_{z^{(i)}=1}^{k} p(x^{(i)}|z^{(i)}; \mu, \Sigma) p(z^{(i)}; \varphi)$. Then, each instance of a search query in a tweet is labeled with an 'influencer' level based on finding the most likely Gaussian distribution $N(\mu_j, \Sigma_j)$ that it came from. The expected influencer level of a user for a given topic is then defined as:

$$usrTpcInflLvl(usr, tpc) = \sum_{twt \in twts(usr)} \sum_{qry \in qrys(twt)} twtQryInflLvl(twt, qry) * tpcQryPrb(tpc, qry)$$

where $tpcQryPrb$ corresponds to φ in the earlier formulation of LDA.

In a real-world setting, it may be desirable to detect influential tweets about a topic in real-time (e.g. before the tweet receives any engagement). To that end, a multi-class SVM is built to predict whether a newly-posted tweet will be influential for each top search query contained in the tweet. A SVM is a supervised learning algorithm that generates a linear binary classifier of the form $h_{w,b}(x) = g(w^T x + b)$, where $g(z) = 1$ if $z \geq 0$ and -1 otherwise. The parameters w and b are found by solving the following optimization problem: $min_{\gamma,w,b} \frac{1}{2}||w||^2 \; s.t. \; y^{(i)}(w^T x^{(i)} + b) \geq 1 \; \forall \, i \in \{1, ..., m\}$.
To build a multi-class SVM, a separate one-versus-rest SVM classifier is built for each class, and then for a given data point the class that has the highest margin is chosen. Specifically, a multi-class SVM is built to predict $twtQryInflLvl(twt, \; qry)$ (e.g. $z^{(i)}$). For training, each instance of a top search query in a tweet is used to generate a training example with the following features:

- User topic probability (from θ) based on the most likely topic for the search query (from ϕ).
- Maximum topic search query probability (from ϕ).
- Average influencer level for the user and search query: $\frac{1}{|twts(usr, \; qry)|} \sum_{twt \in twts(usr, \; qry)} twtQryInflLvl(twt, \; qry)$
- User topic influencer level (from $usrTpcInflLvl$) based on the most likely topic for the search query.

5. Experiments / Results

Perplexity can be used to evaluate the coherence of an LDA model. A lower perplexity value indicates a better representation of the latent topics. Perplexity is defined as: $p(\overline{W}|M) = exp(-\frac{\sum_{i=1}^{m} log(p(\overline{w_i}|M))}{\sum_{i=1}^{m} n_m})$. where M is our model, $\overline{w_i}$ is the search query vector for user i, and n_m is the number of search queries contained in user i's tweets. First, to reduce the number of variables, $\alpha = 1$ and $\beta = 1$ are chosen based on manual inspection.To find the optimal number of topics, for each number of topics, the data is split into 10 parts, one part is held out, the rest of the data is used to train an LDA model, and then perplexity is computed on the held out part. The model with the lowest average perplexity is chosen.

Topics	5	10	15	20	30	50
Ave Perplexity	102.4284	**85.4584**	108.9275	100.5930	90.4831	89.6003

Figure 8: average perplexity for various number of topics

The number of topics that minimizes the average perplexity based on 10-fold cross validation is 10. The next step is to remove search queries that are among the top n most likely words in at least m topics. Based on manual inspection, $n = 20$ and $m = 3$ works well for this dataset, and removes the following queries: "christmas", "exo", "halloween", "likes", and "candy crush". Figure 9 below summarizes the final LDA model.

Topic	NFL	Election	World Series	Tech / Games	UK News
Top Search Queries	Greg Hardy Ken Whisenhunt Andrew Luck Vernon Davis Steve Smith	Trump Ben Carson West Point Bernie Sanders Fred Thompson	Royals Mets #worldseries Harvey Jonny Gomes	#websummit Star Trek Black Ops 3 Warcraft Fallout 4	Sharm John Lewis #bonfirenight #millionmaskmarch Guy Fawkes

Figure 9: top 5 search queries for the top 5 topics outputted by LDA

Generally, the model accurately groups search queries around specific topics or events. One instance of it partially breaking down is when multiple events happen around the same time in the same geographical area, which causes search queries about distinct events to be grouped together. For example, "Sharm", "John Lewis", and "Guy Fawkes" are are all about distinct events but happened around the same time and relate to the UK.

The Akaike Information Criteria (AIC) is used to measure the quality of a statistical model for a given dataset. It is defined as follows: $AIC = 2k - 2ln(L)$, where k is the number of parameters of the model, and L is the maximum value of the likelihood function for the model. The number of influencer levels for our mixture of Gaussians model is chosen

to minimize average AIC across search queries. Note that one value is chosen globally rather than one per-mixture-model (e.g. per search query) so that influencer levels can be compared across search queries.

Influencer Levels	2	3	4	5	10	15	30
Average AIC	9.0940e+04	7.3179e+03	**7.2045e+03**	7.7007e+03	7.3275e+03	7.2868e+03	7.2800e+03

Figure 9: average AIC for a variety of influencer levels

The number of influencer levels that minimizes average AIC is 4. After generating the MLE estimates for φ, μ , and Σ , the allfitdist Matlab toolbox [6] is used to verify that the most likely distribution for engagements given each influencer level is approximately Gaussian, an assumption made in section 4. Figure 10 below summarizes the most influential users for the top 5 topics:

Topic	NFL	Election	World Series	Tech / Games	UK News
Most Influential Users	SportsDayDFW ProFootballTalk NFL Post Sports Sporting News	The Hill POLITICO CNN Politics Fox News ABC News Politics	Michael Baron The Kansas City Star 41 Action News MLB FOX Sports: ML	Web Summit Irish Examiner Independent.ie GameSpot Greg Murphy	ITV News Sky News Daily Mirror The Independent HuffPost UK

Figure 10: most influential users for the top 5 topics

Generally, the model accurately finds accounts that post engaging content specific to the topic and are popular on Twitter. The algorithm finds almost all professional journalists and news outlets, however, and very few non-news accounts which may be better at producing niche content.

To evaluate the multi-class SVM model for predicting an influencer level for a newly posted-tweet and search query, the earliest 90% of tweets are used for training and the rest of the tweets for testing. A confusion matrix and precision and recall are shown in figure 11 and 12 below.

	Level 1	Level 2	Level 3	Level 4
Level 1	1218	178	86	33
Level 2	219	414	67	3
Level 3	143	124	529	17
Level 4	68	40	117	228

Figure 11: confusion matrix for influencer level using all features

Features (incremental)	**Precision**	**Recall**
Ave user query influencer level	64.76%	57.12%
User topic prob	64.89%	57.53%
Max topic query prob	65.1%	58.07%
User topic influencer level	**69%**	**63.67%**

Figure 12: precision and recall incrementally adding features

As shown in figure 12, the average historical influencer level for a user and a query is the best feature for predicting the influencer level for a new tweet and query by that user. Adding in the user topic probability and max topic query probability have a negligible impact on both precision and recall. Adding in the user topic influencer level, however, leads to a sizable increase in precision (4%) and recall (5%). This is due to the fact that in many cases we don't have very many examples of engagements a user received for a specific search query, but have more examples when considering all search queries with a similar topic distribution.

6. Conclusion

In this paper, we explored the problem of building a fine-grained, dynamic topic taxonomy of Twitter interests as well as identifying influential users within each topic. We presented a novel approach of using search queries to define the base entities for LDA, which vastly reduces noise over previous work. Additionally, we presented a new way to combine LDA with tweet engagements to discover influential users for a topic as well as to predict in real-time whether new tweets by an author will be influential for that topic. Relevant future work includes using the follow graph to better detect influencers as well as building a streaming pipeline to augment the topic taxonomy in real-time.

7. References

[1] Hong, Liangjie, and Brian D. Davison. "Empirical study of topic modeling in twitter." *Proceedings of the First Workshop on Social Media Analytics*. ACM, 2010.

[2] Ramage, Daniel, Susan T. Dumais, and Daniel J. Liebling. "Characterizing Microblogs with Topic Models." *ICWSM* 10 (2010): 1-1.

[3] Vosecky, Jan, et al. "Dynamic multi-faceted topic discovery in twitter."*Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*. ACM, 2013.

[4] Weng, Jianshu, et al. "Twitterrank: finding topic-sensitive influential twitterers." *Proceedings of the third ACM international conference on Web search and data mining*. ACM, 2010.

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

[6] Sheppard, Mike. Fit all valid parametric probability distributions to data. Matlab file exchange, 2012.