Identifying Topical Authorities in Microblogs

Aditya Pal*
Department of Computer Science
University of Minnesota
Minneapolis, MN 55455, USA
apal@cs.umn.edu

Scott Counts
Microsoft Research
One Microsoft Way
Redmond, WA 98052, USA
counts@microsoft.com

ABSTRACT

Content in microblogging systems such as Twitter is produced by tens to hundreds of millions of users. This diversity is a notable strength, but also presents the challenge of finding the most interesting and authoritative authors for any given topic. To address this, we first propose a set of features for characterizing social media authors, including both nodal and topical metrics. We then show how probabilistic clustering over this feature space, followed by a within-cluster ranking procedure, can yield a final list of top authors for a given topic. We present results across several topics, along with results from a user study confirming that our method finds authors who are significantly more interesting and authoritative than those resulting from several baseline conditions. Additionally our algorithm is computationally feasible in near real-time scenarios making it an attractive alternative for capturing the rapidly changing dynamics of microblogs.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*information filtering, retrieval models, selection process*; H.3.1 [**Information Systems**]: User/Machine Systems—*human factors, human information processing*

General Terms

Algorithms, Experimentation, Human Factors

Keywords

Microblogging, Twitter, Authority, Clustering, Ranking

1. INTRODUCTION

Users of social media have been called "prosumers" to reflect the notion that the consumers of this form of media are also its content producers. Especially in microblogging contexts like Twitter, for any given topic, the number of these content producers even in a single day can easily reach tens of thousands. While this large number can generate notable diversity, it also makes finding the true authorities, those generally rated as interesting and authoritative on a given topic, challenging. For example, if one wants to get up to speed or stay current on a news story like the *Gulf of Mexico oil spill* that happened over the summer of 2010, who's content should they read? How can we identify those people automatically and reliably for any given topic?

Despite the important role authors serve as *signals* in microblogging, this challenge of identifying true authorities is trickier than it appears at first blush. Perhaps the most important nuance in the discovery of topical authorities is avoiding overly general authorities that typically are highly visible in the network of users because of extremely high values on metrics like the *follower count*. As example, consider again the topic *oil spill*, which is part of the larger category of "news". Top news outlets such as CNN and BBC are authoritative but do not author exclusively or even primarily on this topic and thus recommending only these users is suboptimal. Instead, end users likely are looking for a mix that includes these larger organizations along with lesser known authors such as environmental agencies and organizations, or even the environment departments of the larger news organization in the case of the oil spill topic.

Furthermore, authors may not even exist prior to an event and thus while highly authoritative, they are less discoverable due to low network metrics like the follower count and amount of content produced to date. Consider the Haiti earthquake for which Twitter authors such as *Haiti Earthquake* and *Haiti Relief Funds* are event specific and contributed consistent and detailed content on the event. Due to these rapidly changing dynamics of users in microblogging sites, traditional algorithms based on PageRank over the follower graph of users are sensitive to celebrities and insufficient to find true authorities. Additionally, graph based algorithms are computationally infeasible for near real time scenarios.

We propose an algorithm that alleviates these shortcomings first by incorporating a number of metrics that account for the "topical signal" of both the user and the user's 1-hop network in the follower graph. For example, we compute a self-similarity score that identifies how similar an author's post was to her previous posts. Low scores on this metric

*This research was performed while visiting Microsoft Research.

WSDM'11, February 9–12, 2011, Hong Kong, China.

eliminate authors who post on a wider swathe of topics than just the topic of interest, whereas high scores on this metric eliminate spammers. Rather than using network analysis techniques that might be skewed by disproportionately active or popular users, we further propose using probabilistic clustering to identify a small set of authors with a desirable configuration across our set of proposed metrics. This yields a cluster of largely authoritative authors, who can then be ranked. The full list of proposed metrics and details of our clustering and ranking procedures are presented in Section 3 and 4, respectively. Our algorithm can run in near real time and is thus appropriate for large scale social media environments.

Results of a user study with our method show that it yields significantly better results than several baseline conditions (Section 6). Finally, we show the effectiveness of the probabilistic clustering over other clustering alternatives. Thus the main contribution of this paper is a new method for distinguishing microblogging authors of high topical value that incorporates a number of novel author metrics, utilizes clustering rather than a graph analysis approach to finding authorities, and works efficiently (and effectively) for large scale datasets.

2. RELATED WORK

Within the microblogging arena, little work has explored the issue of authority identification. The notable exception is TwitterRank, proposed by Jianshu et. al. [20], which computes the topical distribution of a user based on Latent Dirichlet Allocation [2] and constructs a weighted user graph where edge weight indicates the topical similarity of the two users. They run a variant of the PageRank [4] algorithm over the directed weighted graph, running the algorithm separately for each topic in order to find authorities on each topic.

While somewhat similar to TwitterRank, our method differs in important ways. First, we incorporate a number of additional features of authors, such as the aforementioned self-similarity score. Second, clustering offers the potential advantage over network-based calculations like PageRank in that it is less prone to skew by a few users with scores that are orders of magnitude larger than the majority of the graph (i.e., celebrities). In fact, a clustering approach might even eliminate unwanted celebrities that do not share enough additional characteristics with other authorities. Finally, but importantly, our method is computationally feasible in near real-time scenarios, making it an attractive choice for capturing the rapidly changing dynamics of microblogs.

Outside microblogging, finding authoritative users in online services generally has been widely studied, with several algorithms proposed towards this goal. Amongst the most popular graph based algorithms are PageRank (Page et. al. [4]), HITS (Klienberg et. al [14]) and their variations. For example, Farahat et. al. [7] proposed a model that combined social and textual authority and defined authority rank based on the HITS algorithm for finding authoritative hyperlinks on the World Wide Web.

Historically, these and other approaches have been applied to domains that far predate microblogging. Social network analysis of Usenet posters revealed the presence of key authors deemed "answer people" (Fisher et. al. [8]). This analysis used nodal features to find users with high out degree and low in degree, under the assumption that those that reply to many, but are rarely replied to are those providing the answers to the questions of the community. Also predating microblogging, several efforts have attempted to surface authoritative bloggers. Java et. al. [10], applying models proposed by Kempe et. al. [13], model the spread of influence on the Blogosphere in order to select an influential set of bloggers which maximize the spread of information on the blogosphere. Java [11] proposed methods to find "blog feeds that matter" using their folder names and subscriber counts.

Authority identification has also been explored extensively in the domain of community question answering (CQA), with several models proposed. As an example of a network modeling approach, Eugene et. al. [1] used the Community Question Answering dataset and extracted several graph features such as the degree distribution of users and their PageRank, hubs and authority scores to model a user's relative importance based on their network ties. They also take into account textual features of the question and answers using KL-divergence between the language model of the two texts, their non-stop word overlap, and the ratio between their lengths.

Others have modeled CQA as a graph induced as a result of a users' interactions with other community members [12, 21]. Zhang et. al. [21] modeled CQA as an expertise graph and proposed *Expertise Ranking*, similar to PageRank [4]. Jurczyk et. al. [12] identified authorities in Q&A communities using link analysis by considering the induced graph from interactions between users.

Still other approaches examined the overall characteristics of users' interactions such as the number of answers, number of questions, number of best answers, votes, and so on. Bouguessa et. al. [3] proposed a model to identify authoritative actors based on the number of best answers provided by users, while Pal et. al. [17] distinguish experts based on their preference in answering position and show that experts are more selective than regular users. Zhang et. al. [21] proposed a measure called *Z-score* that combines the number of answers and questions given by a user to a single value in order to measure the relative expertise of a user. The Z-score measure is based on the intuition that experts generally provide a lot more answers than questions. Extending this approach, topic based models to identify appropriate users to answer a question has been recently proposed by Jinwen et. al. [9].

Summarizing related work, the notion of authority finding has been explored extensively in other domains and has been dominated by network analysis approaches, often in conjunction with textual analysis. Most of these algorithms are computationally expensive. Our domain of interest, microblogging, has seen far less attention, with TwitterRank the notable effort to date. As mentioned above, we feel our approach extends research in the domain by proposing a number of new user metrics and by taking a clustering approach that is computationally tractable to run in near real time scenarios.

3. USER METRICS IN MICROBLOGS

In this and the next section we describe our method for finding authorities. To start we present the list of metrics extracted and computed for each potential authority (see Table 1). Given the nature of tweets (e.g., short text snippets, often containing URLs) and the way they are often

used (for light conversation via replies and for information diffusion via re-tweeting), we focus on metrics that reflect the impact of users in the system, especially with respect to the topic of interest.

We categorize tweets into three categories:*Original tweet* (OT), *Conversational tweet* (CT), *Repeated tweet* (RT).

OT: These are the tweets produced by the author that are not RT or CT.

CT: Conversational tweet is directed at another user, as denoted by the use of the @username token preceding the text or from the meta-data available through the Twitter API.

RT: These tweets are produced by someone else but the user copies, or forwards, them in-order to spread it in her network. These tweets are preceded by "RT @username".

Additionally we compute metrics around the mentions of a user (M) as well as their graph characteristics (G). See Table 1 for the full list of metrics. Most of the metrics are self-explanatory, but we briefly touch upon some of them here. A user can mention other users using the "@user" tag. The first mention in CT and RT is part of the semantic header, so we discard the first mention in these two cases to accurately estimate the genuine mentions that an author makes. Hashtag keywords ($OT4$) are words starting with the # symbol and are often used to denote topical keywords in Twitter.

The self-similarity score ($OT3$) reflects how much a user borrows words from her previous posts (on topic and off topic). In order to compute this score, we first use a stop word list to remove common words and then consider the resulting tweets as a set of words. Removing the common words makes self-similarity more robust. The self-similarity score $S(s1, s2)$ between two sets of words $s1$, $s2$ is defined as:

$$S(s1, s2) = \frac{|s1 \cap s2|}{|s1|} \tag{1}$$

The self-similarity score S is not a true metric because it is asymmetric: $S(x, y) \neq S(y, x)$. We chose this similarity score as it is efficient to compute and because we wish to estimate how much a user borrows words from her previous posts. Also, the restricted character length of microblogs does not lead to large variations in number of words per tweet, so a tf-idf [18] based normalization followed by cosine similarity would not be most effective [15].

In order to compute the self-similarity score for an author, we average similarity scores for all temporally ordered tweets.

$$S(a) = \frac{2 \cdot \sum_{i=1}^{n} \sum_{j=1}^{i-1} S(s_i, s_j)}{(n-1) \cdot n} \tag{2}$$

In equation 2, we assume that the n tweets of an author a are ordered based on increasing timestamp values, s.t., $time(s_i) < time(s_j) : \forall i < j$. A high value of S indicates that user borrows a lot of words or hyperlinks from her previous tweets (suggesting spam behavior). A small value indicates that the user posts on a wider swathe of topics or that she has a very large vocabulary. The self-similarity score is beneficial in our case as we extracted topics based on simple keyword matching, which might lead us to miss related tweets not containing the exact keywords. S ensures that such a similarity is established based on co-occurring terms. A more sophisticated Latent Semantic Analysis [6] approach would lead to computationally expensive and a non-real time performance of the overall algorithm.

ID	Feature
$OT1$	Number of original tweets
$OT2$	Number of links shared
$OT3$	Self-similarity score that computes how similar is author's recent tweet w.r.t. to her previous tweets
$OT4$	Number of keyword hashtags used
$CT1$	Number of conversational tweets
$CT2$	Number of conversational tweets where conversation is initiated by the author
$RT1$	Number of retweets of other's tweet
$RT2$	Number of unique tweets ($OT1$) retweeted by other users
$RT3$	Number of unique users who retweeted author's tweets
$M1$	Number of mentions of other users by the author
$M2$	Number of unique users mentioned by the author
$M3$	Number of mentions by others of the author
$M4$	Number of unique users mentioning the author
$G1$	Number of topically active followers
$G2$	Number of topically active friends
$G3$	Number of followers tweeting on topic after the author
$G4$	Number of friends tweeting on topic before the author

Table 1: List of metrics of potential authorities. OT = Original tweets, CT = Conversational tweets, RT = Repeated tweets, M = Mentions, and G = Graph Characteristics.

3.1 Feature List

We combine the metrics in table 1 to create a set of features for each user. For a given user, we extract the following textual features across their tweets on the topic of interest:

$$\text{Topical signal } (TS) = \frac{OT1 + CT1 + RT1}{|\#\text{ tweets}|} \tag{3}$$

TS estimates how much an author is involved with the topic irrespective of the types of tweets posted by her. Another factor we consider here is the originality of author's tweets, which is calculated as follows:

$$\text{Signal strength } (SS) = \frac{OT1}{OT1 + RT1} \tag{4}$$

SS indicates how strong is author's topical signal, such that for a true authority this value should approach 1. Additionally, we consider how much an author posts on topic and how much she digresses into conversations with other users:

$$\text{Non-Chat signal } (\bar{C}S) = \frac{OT1}{OT1 + CT1} + \lambda \frac{CT1 - CT2}{CT1 + 1} \tag{5}$$

The intuition behind this formulation of $\bar{C}S$ is that we aim to discount the fact that the author did not start the conversation but simply replied back out of courtesy. This can

be desirable when we wish to find real people (i.e. not formal organizations) who are somewhat more social. Since we want $\bar{C}S < \frac{OT1}{OT1+CT2}$, we can solve for λ, by putting this constraint in equation 5 to get:

$$\lambda < \frac{OT1}{OT1 + CT2} \cdot \frac{CT1 + 1}{OT1 + CT1} \tag{6}$$

Empirically, $\lambda \approx 0.05$ satisfies the above constraint for most users in our dataset. Large values of λ can skew the ranking towards real and socially active people whereas small value does not. We do not go any further than this in estimating the effect of λ on model performance.

We compute the impact of an author's tweet by considering how many times it has been retweeted by others:

$$\text{Retweet impact } (RI) = RT2 \cdot \log(RT3) \tag{7}$$

RI indicates the impact of the content generated by the author. This definition of $RT3$ ensures that we dampen the impact for an author who has few overzealous users retweeting her content a lot of times. Note that here we consider $0 \cdot \log(0) = 0$ as the corner case because $RT3 = 0 \Leftrightarrow RT2 = 0$.

In order to consider how much an author is mentioned with regards to the topic of interest, we consider the mention impact of the author as follows:

$$\text{Mention impact } (MI) = M3 \cdot \log(M4) - M1 \cdot \log(M2) \tag{8}$$

MI is based on a similar formulation as that of RI with the difference that we take into account a factor that estimates how much the author mentions others. This ensures that we incorporate the mentions an author receives purely based on her merit and not as a result of her mentioning others.

In order to estimate how much influence is diffused by the user in her network, we take into account the following feature:

$$\text{Information diffusion } (ID) = \log(G3+1) - \log(G4+1) \tag{9}$$

ID is the ratio of number of users activated by the author and the number of users that activated the author on log-scale. Here "activated" means tweeting on a topic after another user from the user's network has tweeted on topic before the author. We add 1 in this case and rest other cases in order to avoid a divide by zero operation. This adheres to the *rule of succession* as proposed by Laplace. Note that ID does not take into consideration the advantage an author with large in-degree and low out-degree might have, namely that $G3$ can be a large value whereas $G4$ remains bounded by a small number of friends. For such a case to occur an author must be amongst the early publishers on the topic, which is a sign of authoritativeness. We experimented with alternate formulations of ID,

$$ID_1 = \log(\frac{G3+1}{G1+1}) - \log(\frac{G4+1}{G2+1}) \tag{10}$$

ID_1 normalizes ID on raw count of topical followers and friends. This formulation leads to less effective results than the un-normalized version. One reason is that it fails to capture the prominence of a person as indicated by the raw counts itself, hence we do not consider the alternate formulation of ID any further.

Additionally, we consider the raw number of topically active users around the author, as follows:

$$\text{Network score } (NS) = \log(G1+1) - \log(G2+1) \tag{11}$$

In all these cases, we consider log scaling around the network parameters because the underlying distribution of network properties follows a tail distribution with some users with orders of magnitude larger metric values than others. This could lead to skew while clustering.

It should be noted that all these features are fairly straightforward to compute given an author's tweets and one hop network. Additionally these features can be computed in parallel for all the users making it fit to run in a MapReduce [5] type distributed computing framework.

4. CLUSTERING AND RANKING

We used a Gaussian Mixture Model to cluster users into two clusters over their feature space. The main motivation for the clustering was to reduce the size of the target cluster (i.e., the cluster containing the most authoritative users). This also makes the subsequent ranking of users more robust because it is less sensitive to outliers such as celebrities. The following subsection describes Gaussian Mixture Modeling in general and how we used it in our setting.

4.1 Gaussian Mixture Model

Clustering based on Gaussian mixture model is probabilistic in nature and aims at maximizing the likelihood of the data given k Gaussian components. Consider n data points $\mathbf{x} = \{x_1, x_2, ..., x_n\}$ in d-dimensional space, the density of any given data point x, can be defined as follows:

$$p(x|\pi, \Theta) = \sum_{z=1}^{k} p(z|\pi) \cdot p(x|\theta_z) \tag{12}$$

where π is the prior over the k components and $\Theta = \{\theta_z : 1 \leq z \leq k\}$ are the model parameters of the k Gaussian distributions i.e. $\theta_z = \{\mu_z, \Sigma_z\}$ and $P(x|\theta_z)$ is defined as:

$$p(x|\theta_z) = \frac{1}{((2\pi)^d |\Sigma_z|)^{\frac{1}{2}}} \exp\{-\frac{1}{2}(x-\mu_z)^T \Sigma_z^{-1} (x-\mu_z)\} \tag{13}$$

Under the assumption that the data points are independent and identically distributed (i.i.d), we can consider the likelihood of the observed samples as follows:

$$p(\mathbf{x}|\pi, \Theta) = \prod_{i=1}^{n} P(x_i|\pi, \Theta) \tag{14}$$

$$= \prod_{i=1}^{n} \sum_{z=1}^{k} p(z|\pi) \cdot p(x_i|\theta_z) \tag{15}$$

In order to maximize this likelihood, we use Expectation Maximization (EM). EM is an iterative algorithm in which each iteration contains an E-step and a M-step. In the E-step, we compute the probability of the k Gaussian components given the data points, $p(z|x_i, \Theta)$ using Bayes theorem:

$$p(z|x_i, \pi, \Theta) = \frac{p(x_i|\theta_z) \cdot p(z|\pi)}{\sum_{z=1}^{k} p(x_i|\theta_z) \cdot p(z|\pi)} \tag{16}$$

In the M-step, we compute the model parameters in order to maximize the likelihood of the data, as follows:

$$\mu_z = \frac{\sum_{i=1}^{n} x_i \cdot p(z|x_i, \pi, \Theta)}{\sum_{i=1}^{n} p(z|x_i, \pi, \Theta)} \tag{17}$$

$$\Sigma_z = \frac{\sum_{i=1}^{n} (x_i - \mu_z) \cdot (x_i - \mu_z)^T \cdot p(z|x_i, \pi, \Theta)}{\sum_{i=1}^{n} p(z|x_i, \pi, \Theta)} \tag{18}$$

$$p(z|\pi) = \frac{\sum_{i=1}^{n} p(z|x_i, \pi, \Theta)}{\sum_{z=1}^{k}\sum_{i=1}^{n} p(z|x_i, \pi, \Theta)} \quad (19)$$

The EM algorithm is run iteratively until the likelihood reaches the maximum possible value. The GMM model requires initial estimates of the model parameters θ and prior probability of the components $p(z|\pi)$. We use K-means to derive these initial parameters. In general, GMM performs better than classical hard clustering algorithms such as K-means as it is less sensitive to outliers. A drawback of GMM is that it is sensitive to initial estimates of model parameters. This problem can be eliminated by running GMM with boosting [19]. Since, this can be computationally expensive, we simply run 5 instances of GMM (with maximum of 50 iterations each) and pick the one with largest log likelihood as the best estimate of the underlying clusters.

The above clustering algorithm gives probabilistic assignments of data points belonging to a given cluster $p(z|x, \pi, \Theta)$. For each cluster, we pick all the points with this probability to be greater than 0.9. This is done as we want the true representative points per cluster. Using these points, we compute the average TS, RI, MI per cluster and pick the cluster with the larger TS, RI, MI (or best of 3) as our target cluster. This simple strategy of determining which cluster to pick works well in practice. We explored computing the centroid of the clusters and picking the cluster farthest away from the origin using Mahalanobis or Euclidean distance, but found that approach agreed with our heuristic (best of 3) every time.

The target cluster typically contains a small number of users (a few hundred to thousands) which is a huge reduction compared to the actual number of users (ranging from tens of thousands to hundreds of thousands). In order to rank authors within the target cluster, we explored two potential methods: List based ranking and Gaussian based ranking. In order to describe these ranking methods, consider that we have n data points $x = \{x_1, x_2, ..., x_n\}$ where each data point is a d-dimensional vector, i.e., $x_i = [x_i^1, x_i^2, ..., x_i^d]^T$. In list based ranking, we sort authors on feature $f \in \{1, 2, ..., d\}$ and get the rank of i^{th} author in this ranked list, denoted as $R_L(x_i^f)$. The final rank of an author is the sum of ranks for all the features, $R_L(x_i) = \sum_{f=1}^{d} R_L(x_i^f)$, which is then used to sort the authors to get a ranked list. Assuming we want top k authors, this results in time complexity of $O(dn \log n)$. Ultimately we found this list based approach inferior to Gaussian ranking method that we used, which is described in the next section.

4.2 Gaussian Ranking Algorithm

We assume features to be Gaussian distributed (which is true in most case, though with a bit of skew in some cases). For any given feature f, we compute the μ_f and σ_f based on the points in the target cluster. The Gauss rank R_G of a given data point is then defined as follows:

$$R_G(x_i) = \prod_{f=1}^{d} \int_{-\infty}^{x_i^f} N(x; \mu_f, \sigma_f) \quad (20)$$

where $N(x; \mu_f, \sigma_f)$ is the univariate Gaussian distribution with model parameters as μ_f and σ_f. The inner integral in this equation computes the Gaussian cumulative distribution at x_i^f. Gaussian CDF is a monotonically increasing function well suited to our ranking problem as we prefer a higher value over low value for each feature. Alternately, if a low value is preferred for some features, then x_i^f could be replaced by $-x_i^f$ in the above formula. Gaussian CDF (for standard normal) can be computed in $O(1)$ time using a pre-computed table of error functions. This results in an algorithm with time complexity of $O(dn + k \log k)$ which is a reduction by a factor of $\log n$ for small k over List based ranking.

Additionally, we explored a weighted version of Gaussian ranking method. In order to incorporate weights in the above Gaussian ranker, we consider the following definition of R_G:

$$R_G(x_i) = \prod_{f=1}^{d} [\int_{-\infty}^{x_i^f} N(x; \mu_f, \sigma_f)]^{w_f} \quad (21)$$

where w_f is the weight that we put on feature f. These ranks help in devising a total ordering under $\leq$ over all the users in the target cluster. Using this fact, we observe that the weights w_f are immune to the normalization factor (as long as the normalization factor is greater than 0). In this case the normalized rank (with normalization factor N) is $\acute{R}_G = R_G^{\frac{1}{N}}$, which doesn't change the ordering of data points for $N > 0$. Hence, the only constraint we put on these weights is that $\{\forall f : 0 \leq w_f \leq 1\}$.

5. DATASET

To serve as test data for our experiments, we collected all tweets posted on Twitter between 6th-June-2010 to 10th-June-2010 (5 days overall). These were available through access to the full Twitter dataset (the "firehose") granted to our company. The dataset consists of 89,622,039 tweets. We extracted tweets on three topics: *oil spill*, *world cup* and *iphone* using simple substring matching. Note that we could take a LDA [2] type approach on the tweets extracted based on string matching to find other tweets with similar latent topical distribution which would be a more comprehensive corpus for the given topic. That is extremely resource consuming and could take days to complete on the scale of data, we had. Table 2 presents the basic statistics of the extracted topical data. The in-degree distribution of the users in our dataset follows a Pareto distribution with a slight skew. Due to space constraints we skip a detailed description of the dataset.

	$\|U\|$	$\|OT\|$	$\|CT\|$	$\|RT\|$
iphone	430,245	658,323	242,000	129,560
oil spill	64,892	111,000	8,140	29,224
world cup	44,387	308,624	28,612	47,837

Table 2: Dataset statistics. $|U|$, $|OT|$, $|CT|$, $|RT|$ are overall count of users, original tweets, conversational tweets and retweets, respectively.

6. RESULTS AND EVALUATION

We compared our model with several baseline models as described below:

our: Our model as based on the features described in section 3.1. Additionally, we use $\frac{OT2}{OT1}$, $OT3$, $\frac{OT4}{OT1}$. Based on these features, we run the methods as described in Section 4.

iphone	oil spill	world cup
macworld	NWF	TheWorldGame
Gizmodo	TIME	GrantWahl
macrumorslive	huffingtonpost	owen_g
macTweeter	NOLAnews	guardian_sport
engadget	Reuters	itvfootball
parislemon	CBSNews	channel4news
teedubya	LATenvironment	StatesideSoccer
mashable	kate_sheppard	Flipbooks
TUAW	MotherNatureNet	nikegoal
Scobleizer	mparent77772	FIFAWorldCupTM

Table 3: List of top 10 authors for the three topics as computed by our algorithm.

	our	*b*1	*b*2	*b*3
iphone	282665	**1364015**	117250	1252
oil spill	462507	**871159**	417210	840
world cup	29373	**32121**	18017	277

Table 4: Average number of followers for the top 10 authors of various algorithms.

b1: This model consists of graph properties: RI, MI, ID, NS. Additionally, we considered page rank as a dimension of user's feature vector. In order to compute page rank we created directed weighted mention graph where an edge from x to y indicates "how many times x mentions y, averaged for all out links of x". Typically this results in several disconnected components. We computed page rank on this graph with a teleport probability of 0.15 (which ensures that the Markov chain reaches stationary distribution after sufficient iterations resulting in convergence of the algorithm). The clustering and ranking algorithm used in our method is then applied to construct a list of top 10 users.

b2: This model consists of the textual properties of the users: TS, SS, $\bar{C}S$, $\frac{OT2}{OT1}$, $OT3$, $\frac{OT4}{OT1}$. Our clustering and ranking algorithm is then applied to construct a list of top 10 users.

b3: In this model, authors that fall outside the target cluster are randomly selected. This model helps in validating our target cluster selection criteria.

6.1 Top 10 Ranked Authors

To give a sense of how well our algorithm works, Table 3 presents the list of top 10 users as recommended by our algorithm. Some of these users are large organizations (*TIME*, *mashable*) yet the list contains a lot of real people that are correspondents of organizations dealing in that topic (*Grant Wahl* for *world cup*) and several small organizations (such as *NWF* and *LATenvironment* for *oil spill*). These real people and smaller organizations are fairly on topic (and relevant) and do not enjoy as high popularity as the topical celebrities do. The algorithm rejected several celebrities as clustering disregards these people on several other dimensions or they are not true representatives of the target cluster (probability ≥ 0.9). For other topics, for example, toy story 3 our algorithm returned leeunkrich (the director of the movie) as the top user, while rejecting celebrities who tweeted about the movie. Table 4 shows that the average number of followers for our model is lesser than b1 and higher than b2 indicating that it strikes a balance between network and textual properties of the users that influence the topic.

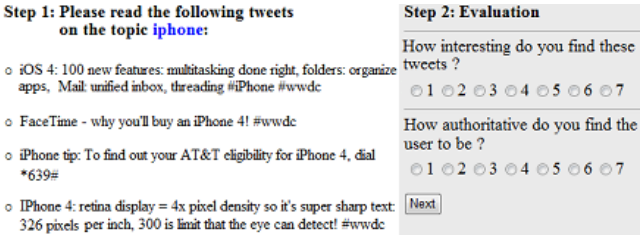

Figure 1: Anonymous survey screen shows four topical tweets of an author and asks evaluators to rate for *Interestingness* and *Authoritativeness* on the scale of 1-7 (7 being the highest).

Figure 2: Non-anonymous survey screen shows name and four topical tweets of an author and asks evaluators to rate for *Interestingness* and *Authoritativeness* on a scale of 1-7 (7 being the highest).

6.2 Model Rating Comparison

In order to evaluate our approach, we conducted a user study in which results from our model were compared to those from three baseline models across three topics. From our model we selected authors that were in the top 20 for each topic[1]. From the three baseline models, we selected 10 users each. There was enough overlap between the ranked lists of authors produced by the four models that it resulted in 40 authors per topic. Finally, every author evaluation was made both anonymously (such that the name of the author was not shown) and non-anonymously. Thus our experimental design was a 3 (topic) X 4 (ranking method) X 2 (anonymous) design.

Each participant was shown 40 screens, each with a different author. Each screen asked participants to evaluate both the author (and her tweets) on "*How interesting and authoritative they found the author and her tweets*" using two 7-point Likert scales. The first 20 screens prompted for anonymous evaluation (see figure 1) and the next 20 screens prompted for non-anonymous evaluation (see figure 2). Note that the only difference between the anonymous and non-anonymous ratings was that in one case the name of the author was shown, while in the other it was not shown.

We note here our rationale for having authors rated both anonymously and non-anonymously. First, this enabled us to establish a ground truth about the users recommended by our algorithms without any effect of bias due to ratings being

[1]We performed an additional comparison within our model only of those ranked in the top ten versus those in the second ten.

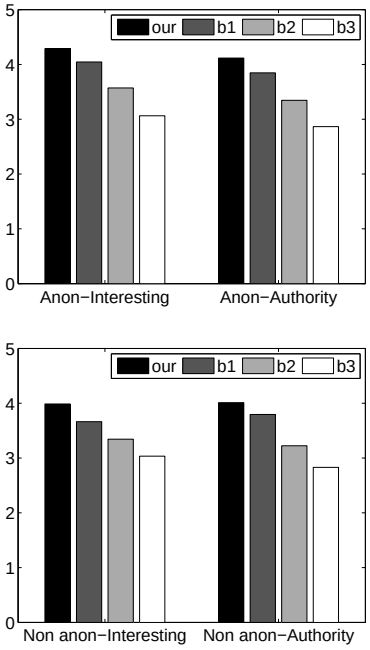

Figure 3: Average ratings per model per participating user. Average is computed by first aggregating ratings received by top 10 authors of each model by each participant. This average is computed across all topics.

made on the status of the author as conveyed by the name rather than on the quality of the content (the anonymous case). Second, we could evaluate authors in a real world way in which user names are known (the non-anonymous case). From a practical standpoint, anonymous ratings may be useful when building a reading list of interesting items for a user, whereas the latter case makes sense when recommending authors for users to follow.

We randomized the order in which the evaluations were shown such that for each evaluated author we had equal numbers of anonymous and non-anonymous ratings. Also, each participant evaluated an author only once either anonymously or non-anonymously. 48 users participated in the survey out of which 25% were female participants and the average age of all the participants was 32.1 (median = 31) with standard deviation of 5.9. On average, we received 16 ratings per evaluated author (8 anonymous and 8 non-anonymous). The inter-rater agreement (Fleiss kappa) between participants was 0.56, which can be considered as moderate agreement between participants.

In order to compare the ratings received by the four models, we computed aggregate ratings given by each respondent to authors of each model. This results in 4 (models) X 4 (response variables) averages per respondent. The four response variables were: *Anonymous - Interesting* (AI), *Anonymous - Authority* (AA), *Non anonymous - Interesting* ($\neg AI$), *Non anonymous - Authority* ($\neg AA$). Figure 3 shows that our algorithm received the highest ratings compared to the baseline models for all the four response variables.

To establish that the authors of our algorithm received statistically significantly higher ratings than other models,

	iphone	oil spill	world cup	**overall**
AI	0.018	0.024	*0.45*	0.002
AA	0.012	0.014	*0.59*	0.001
$\neg AI$	0.003	0.023	0.006	0.001
$\neg AA$	0.021	0.049	0.024	0.021

Table 5: Our vs b1 model. P-value for paired one sided t-test of *average* ratings per model per participant. $H0$ is rejected in all cases except for AI and AA for *world cup*.

	iphone	oil spill	world cup	**overall**
AI	0.001	0.011	*0.084*	0.001
AA	0.001	0.006	*0.5*	0.001
$\neg AI$	0.024	0.001	0.004	0.001
$\neg AA$	0.018	0.044	0.006	0.001

Table 6: Our vs b1 model. P-value for paired one sided t-test of *best* ratings per model per participant. $H0$ is rejected in all cases except for AI and AA for *world cup*.

we compare the aggregate ratings using one-sided paired t-tests with the hypothesis: *H0: Rating means of the two models are the same* and *Ha: Rating mean of our model is higher with 95% Confidence Interval*. Table 5 shows the p-values of the t-test of *our* vs the $b1$ model. We reject $H0$ in all the cases except for the topic *world cup* for AI and AA. Overall, we establish that the average ratings given by respondents to users of our model are higher than the b1 model. We note here that the other two baseline models fared worse than b1, and we thus conclude that our model outperformed all three baseline models.

We also observe from Figure 3 that on average anonymous ratings are higher than the non-anonymous ones. This indicates that respondents are more conservative in giving good ratings when author names are known to them, likely because ratings go down slightly when made on an unrecognized author. Section 6.3 discusses it in detail.

6.2.1 Model Rating Comparison Under Realistic Circumstances

In most realistic circumstances, we envision that when a recommendation engine returns a ranked list to a web user, the web user simply clicks on one of the ranked objects (as in search). Similarly in our case, we can argue that if we return a list of 10 authors out of which 9 are bad but one is extremely good and the user clicks the good author and finds her to be interesting then the user's experience with the recommendation engine is successful and the engine performs well with regards to this user. In order to incorporate this scenario, we consider the ratings given to best rated author per model by each respondent and compare these best ratings rather than comparing the average ratings (as done previously). Table 6 shows that our model receives significantly higher best ratings compared to the $b1$ model (except for the two response variables for *world cup*). Overall, we conclude that our model performs better than all the other baseline models even when considering only the top rated author.

6.3 Anonymous vs Non Anonymous Ratings

Every evaluated author received 8 anonymous and 8 non-

	AI vs $\neg AI$	AA vs $\neg AA$
U_f	0.49	0.84
U_{nf}	0.001 (↓)	0.017 (↓)
U_f (with bad rating)	0.02 (↑)	0.008 (↑)
U_{nf} (with bad rating)	0.77	0.61

Table 7: P-value for paired one sided t-test for U_f and U_{nf} users between anonymous and non-anonymous counterparts of response variables. The up arrows indicate that rating means are higher for the second response variable whereas down arrow indicates that rating means are higher for the first response variable.

anonymous ratings. We computed one sided paired t-tests on aggregated ratings of authors. The p-value between AI and $\neg AI$ is < 0.001 indicating that the anonymous ratings for interestingness are higher than the non-anonymous case. On the other hand the p-value for AA and $\neg AA$ is 0.17 indicating that there is no significant change in authority ratings from the non-anonymous to the anonymous case. This only partially confirms the intuition that respondents get stricter when rating non-anonymously.

We thus considered two buckets of authors: > 50000 followers (U_f - famous authors), ≤ 50000 followers (U_{nf} - non famous authors). Further we can subdivide them based on whether they received good or bad ratings. We define good rating to be ≥ 4. Table 7 summarizes the p-value between the response variables for the several categories of users.

Even though we expected the rating averages to increase for U_f, the p-value fails to indicate a statistically significant change. Our explanation for this is that there is a ceiling effect in place. For famous authors who received good ratings anonymously, these ratings would not change drastically (or consistently for all such authors) once their identity is revealed. On the other hand for the famous authors who received bad ratings in anonymous case (40% - 50%), their ratings would consistently increase when their names are shown (as confirmed by the third row of table 7). As expected non-anonymous ratings for non-famous authors decrease as their names are shown. Overall, we conclude that while unrecognized users may suffer a bit when their names are shown, the popular users are getting a boost in their authority rating simply due to "name value".

6.4 Top 1-10 versus Top 11-20

Next we compare the ratings received by the top 1-10 and the top 11-20 authors recommended by our algorithm. Figure 4 indicates that the average rating of the top 1-10 is higher than that of the top 11-20 for all four response variables. Again using one sided paired t-test, we reject H0 (i.e. rating means are same) for AI and $\neg AA$. We failed to reject the null hypothesis for AA and $\neg AI$ slightly ($p \leq 0.057$). We conclude that the top 10 authors are significantly better than the next 10.

6.5 Model Precision

In order to compute precision and recall values, we sort authors based on their aggregate survey ratings (separately for all 4 response variables) and pick the 10 highest rated authors. Precision can be computed by counting the number of authors that were correctly identified to be in top 10 by the algorithm. Note that in this case recall = precision since

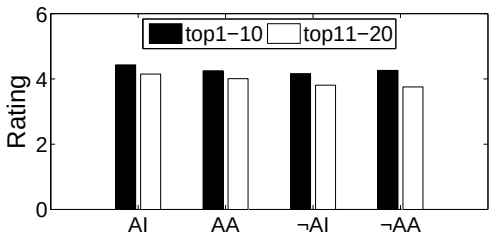

Figure 4: Average rating received by authors of top 1-10 vs top 11 - 20 recommended by our model.

the two list sizes are the same. Table 8 shows the precision of our algorithm (≥ 0.6).

Similarly, we compare our model with other models based on the above idea. In order to do that, we picked authors recommended by the two models that are to be compared. These authors are then sorted and the 10 authors with highest ratings selected. Precision of our model is then computed based on how many authors recommended by it and appear in this top 10 list. Table 9 reports the precision of our algorithm *vs* the two baseline models. So the first row (and first column) indicates that 8 out of 10 users predicted by our model were in top 10 and only 2 from $b1$, which indicates that our model is substantially better than $b1$ (and also $b3$).

	iphone	oil spill	world cup	**overall**
AI	0.8	0.8	0.6	0.73
AA	0.8	0.7	0.5	0.63
$\neg AI$	0.7	0.7	0.6	0.6
$\neg AA$	0.6	0.7	0.6	0.6

Table 8: Absolute precision (or recall) of *our* algorithm.

	our vs $b1$	*our* vs $b3$
AI	0.8	1
AA	0.6	0.93
$\neg AI$	0.73	0.93
$\neg AA$	0.63	0.87

Table 9: Precision (or recall) of *our* algorithm vs $b1$ and $b3$ aggregated for the three topics. While computing precision of *our* vs $b1$, authors that were common in *our* and $b1$ were discarded.

6.6 Algorithm Effectiveness

In order to measure effectiveness of the algorithm, we correlated ordered list of the top 10 authors as recommended by our algorithm with the corresponding top 10 list based on the ratings provided by the survey respondents. Table 10 shows the Pearson correlation[2] of *our* algorithm versus the ratings provided by respondents. We only report the correlations while considering ratings on AI and $\neg AA$ as the criteria to generate ranked list from survey respondents. Overall the Pearson correlation for the two measures is 0.39,

[2]Since the rankings are tied in some cases, Pearson correlation is preferred over Spearman [16]. Additionally, the Spearman values were approximately the same as Pearson in our case.

	our	*our* (Kmeans)	*our* (no clustering)
iphone	0.54	0.40	-0.07
oil spill	0.41	0.29	-0.05
world cup	0.22	0.14	0.06
overall	0.39	0.28	-0.02

Table 10: Pearson correlation of several version of *our* algorithm with the *AI* ratings of survey respondents.

which we consider to be more than satisfactory considering survey respondents as the ground truth.

In a similar fashion, we can run our algorithm with different clustering algorithms and generate the relative ordering of top 10 authors (as given by our algorithm when GMM based clustering is used). We see in table 10 that the GMM based version of *our* model is more closely related to *AI* ratings of respondents than other alternative algorithms (including ranking without clustering). With this we conclude that probabilistic clustering (and clustering in general) is an important step that helps in eliminating outliers in each feature dimension and providing robustness to overall ranking.

Additionally, we evaluated the effectiveness of List based ranking in comparison to Gaussian based ranking. Using the methodology described above, we record the Pearson on both measures to be ≈ 0.17, indicating that the Gaussian based ranking performs much better than the list based.

6.7 Estimating Optimal Weights

We aim to estimate the weight parameters for the weighted version of Gaussian rank (see Equation 21), such that we maximize the Pearson correlation with the respondent's *AI* ratings. For each weight vector w, we run *our* algorithm and measure the Pearson correlation with the survey-based rating list of top authors. This correlation is henceforth called the score of the weight vector w. To find the weights that maximize the score, we use stochastic hill climbing along with simulated annealing in the unit hypercube that encloses all possible weight vectors (recall that weights are bounded by $[0, 1]$). We skip the details of the algorithm used due to space constraints.

Once optimal weights are found per topic, we consider two hypersphere of radius δ and $\delta + 0.01$ around these optimal weights. We sample weights that are enclosed in the larger hypersphere and not in the smaller one and compute the max score of the weight samples drawn. Figure 5 shows the distribution of scores as δ is increased around the optimal weights (ensuring that the weights lie within unit hypercube). Even though the score distribution is very spiky, we see the overall shape looks like a bell shaped curve. This also indicates that the best possible model (given our Model Selection) can at best achieve a correlation of 0.61 (iphone) and 0.71 (oil spill). The weights that maximize scores for both the topics leads to a correlation of 0.56 (iphone) and 0.61 (oil spill). The unweighted model achieves 0.54 (iphone) and 0.41 (oil spill), which is more than satisfactory.

Our overall assessment after manual analysis of the optimal weights lead us to believe that *topical signal* and *mention impact* should be assigned slightly higher weights than other features, though we need to give this a more thorough treatment before generalizing.

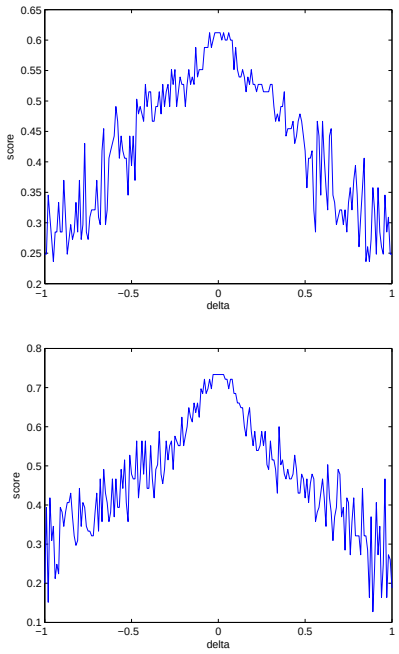

Figure 5: Score around best possible weights for topics *iphone* and *oil spill*.

7. DISCUSSION

The results of the user study confirm that our method for authority finding yielded authors of greater interest and authoritativeness than the baseline comparisons. Here we discuss aspects of the method we used and attributes of the types of authors that might explain this improvement. First, to what extent does the popularity of the author matter when it comes to being interesting and authoritative? At the outset of the paper, we argued that some combination of popular and less popular authors is a likely "sweet spot". For a systematic comparison, we isolated the role that name value of authors plays when evaluating their content. The anonymous and non-anonymous ratings show that anonymous ratings generally were a bit higher, while lower rated, but popular authors get a boost when their names are revealed. From this we conclude that from a perceptual standpoint, popularity matters and again that the ideal set of authors contains those with the highest rated authority regardless of popularity mixed with those who are popular. In other words, give users authors of quality content and also authors they recognize.

The precise balance between popular and less popular authors may depend on both topic and timing. When a topic is pressing and of a certain size (e.g. *iphone*), popular users (such as *mashable*) who don't tweet exclusively on the topic, likely are in fact devoting considerable content to the topic. In other cases, such as *world cup*, top celebrities such as *Shakira* who dominated in terms of retweets and graph characteristics need to be correctly rejected. Our similarity score metric helped make this distinction between which popular users were on topic enough to show in the results set.

Returning to the issue of network versus text based features, it is important to remember that unlike blogging, microblogging is a more dynamic environment, in which the

lifetime of a topic can be very short-lived. In order to find topical authorities in such an environment, a purely graph based approach can wrongly assign a person with authority in some other topic, or simply a celebrity with many followers, to be an authority on the topic of interest. In terms of common nodal characteristics, our model did in fact yield authors with follower counts in between models based on the follower graph and on textual features (Table 4). We do note, however, that the $b1$ baseline model, which was based on graph characteristics, fared considerably better than the model based on textual features ($b2$). With respect to narrowing down the list of top authors, probabilistic clustering appears to be a good choice. In our results, GMM reduced the set of possible authorities from tens of thousands to a few hundred. We also showed that this clustering technique yielded results that correlated more highly with end user ratings than other clustering techniques.

In terms of the final ranking of users, we saw that authors at the top of our results list (top 10) were in fact rated more positively than those just down the list (11-20). This suggests that the Gaussian ranking procedure works well, and we suspect the improvements seen over list-based ranking (see section 6.7) would generalize to other social media contexts. In terms of which features are most important when ranking users, we tentatively conclude that *topical signal* and *mention impact* should be assigned higher weights than other features, though exploring this in greater detail remains an area for future work.

8. CONCLUSION AND FUTURE WORK

In this paper we proposed features and methods that could be used to produce a ranked list of top authors for a given topic for identifying topical authorities in microblogging environments. We proposed a number of features of authors and observe that *topical signal* and *mention impact* are slightly more important than other features.

We also showed that probabilistic clustering is an effective way to filter a large chunk of outliers in the feature space (either long tail or celebrities) and select high authority users on which ranking can be applied more robustly. Finally, we show that Gaussian-based ranking is a more effective and efficient way to rank users. Results of our study shows that our model is better than the baseline models considered, and we emphasize that our model can be used in near real-time scenarios.

For future work, we wish to explore in detail how different weights affect the final author rankings and what weight distribution is most effective for a given problem domain. As an example, we would like to estimate the influence of negative weights on features. We would also like to investigate effective ways to filter large organizations in order to build a more socially oriented people recommender.

9. REFERENCES

[1] E. Agichtein, C. Castillo, D. Donato, A. Gionis, and G. Mishne. Finding high-quality content in social media. In *WSDM*, pages 183-194, 2008.

[2] D. Blei, A. Y. Ng, and M. I. Jordan. Latent Dirichlet Allocation, *JMLR*, 3:993-1022, 2003

[3] M. Bouguessa, B. Dumoulin, and S. Wang. Identifying authoritative actors in question-answering forums: the case of yahoo! answers. In *KDD*, pages 866-874, 2008.

[4] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford Digital Library Technologies Project, 1998.

[5] J. Dean and S. Ghemawat. Mapreduce: Simplified data processing on large clusters. In *OSDI*, pages 137-150, 2004.

[6] S. C. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. Harshman. Indexing by latent semantic analysis. In *Journal of the American Society for Information Science (JASIS)*, 41(6):391-407, 1990.

[7] A. Farahat, G. Nunberg, and F. Chen. Augeas: authoritativeness grading, estimation, and sorting. In *CIKM*, pages 194-202, 2002.

[8] D. Fisher, M. Smith, and H. T. Welser. You are who you talk to: Detecting roles in usenet newsgroups. *Hawaii International Conference on System Sciences (HICSS)*, 3:59b, 2006.

[9] J. Guo, S. Xu, S. Bao, and Y. Yu. Tapping on the potential of Q&A community by recommending answer providers. In *CIKM*, pages 921-930, 2008.

[10] A. Java, P. Kolari, T. Finin, and T. Oates. Modeling the spread of influence on the blogosphere. In *WWW (Special interest tracks and posters)*, 2006.

[11] A. Java, P. Kolari, T. Finin, and T. Oates. Feeds That Matter: A Study of Bloglines Subscriptions. In *ICWSM*, 2007.

[12] P. Jurczyk and E. Agichtein. Discovering authorities in question answer communities by using link analysis. In *CIKM*, pages 919-922, 2007.

[13] D. Kempe. Maximizing the spread of influence through a social network. In *KDD*, pages 137-146. 2003.

[14] J. M. Kleinberg. Authoritative sources in a hyperlinked environment. In *SIAM symposium on Discrete algorithms (SODA)*, pages 668-677, 1998.

[15] D. Metzler, S. T. Dumais, and C. Meek. Similarity measures for short segments of text. In *ECIR*, pages 16-27, 2007.

[16] J. L. Myers and A. D. Well. Research Design & Statistical Analysis. *Routledge Academic*, 2003.

[17] A. Pal and J. A. Konstan. Expert Identification in Community Question Answering: Exploring Question Selection Bias. In *CIKM*, pages 1505-1508, 2010.

[18] K. Sparck Jones. A statistical interpretation of term specificity and its application in retrieval. In *Journal of Documentation*, pages 132-142, 1988.

[19] F. Wang, C. Zhang, and N, Lu. Boosting gmm and its two applications. In *Lecture Notes in Computer Science*, (3541):12-21, 2005.

[20] J. Weng, E.-P. Lim, J. Jiang, and Q. He. Twitterrank: finding topic-sensitive influential twitterers. In *WSDM*, pages 261-270. 2010.

[21] J. Zhang, M. S. Ackerman, and L. Adamic. Expertise networks in online communities: structure and algorithms. In *WWW*, pages 221-230, 2007.