Finding topical experts in Twitter via query-dependent personalized PageRank

Preethi Lahoti
Aalto University
Helsinki, Finland
preethi.lahoti@aalto.fi

Gianmarco De Francisci Morales
Qatar Computing Research Institute
Doha, Qatar
gdfm@acm.org

Aristides Gionis
Aalto University
Helsinki, Finland
aristides.gionis@aalto.fi

***Abstract*—Finding topical experts on micro-blogging sites, such as Twitter, is an essential information-seeking task. In this paper, we introduce an expert-finding algorithm for Twitter, which can be generalized to find topical experts in any social network with endorsement features.**

Our approach combines traditional link analysis with text mining. It relies on crowd-sourced data from Twitter lists to build a labeled directed graph called the *endorsement graph*, which captures topical expertise as perceived by users. Given a text query, our algorithm uses a dynamic topic-sensitive weighting scheme, which sets the weights on the edges of the graph. Then, it uses an improved version of query-dependent PageRank to find important nodes in the graph, which correspond to topical experts. In addition, we address the scalability and performance issues posed by large social networks by pruning the input graph via a focused-crawling algorithm.

Extensive evaluation on a number of different topics demonstrates that the proposed approach significantly improves on query-dependent PageRank, outperforms the current publicly-known state-of-the-art methods, and is competitive with Twitter's own search system, while using less than 0.05% of all Twitter accounts.

I. Introduction

The social Web enables people to stay connected with their friends, but also to be informed about what is happening in the world. While social media provide tremendous opportunities for information seeking, they also pose grand challenges. First, the sheer amount of generated data leads to *information overload*. Second, given that social media empower everyone to be a source of information, finding *authoritative* sources becomes significantly more challenging.

The task of *finding authorities with respect to a given topic* is a fundamental information-seeking operation. Traditionally, the problem of finding topical authorities in networked data has been addressed by combining text mining with link analysis [6, 9, 13, 18]. However, existing methods cannot be easily adapted to microblogging platforms for several reasons. Unlike in other linked structures such as the Web, in microblogging sites when a user *endorses* another user via a "follow" or a "retweet," it is hard to identify the topic that the endorsement refers to, due to the fact that the contextual text is short and noisy.

ASONAM '17, July 31-August 03, 2017, Sydney, Australia

ACM ISBN 978-1-4503-4993-2/17/07 $15.00
http://dx.doi.org/10.1145/3110025.3110044

In addition, inferring the topical interests of users based on generated content and self-biographies is a challenging task, and combining the network structure with the user-generated content is not straightforward.

In this paper, we propose a novel method for finding topical authorities in social networks, and specifically in Twitter. The crux of our approach is to process *Twitter lists*, i.e., manually-curated collections of Twitter accounts, which are created in a crowd-sourced manner, and which typically refer to a single topic. For instance, a Twitter user may create a list named `travel` and include in it accounts such as `@TravelMoments` and `@travel_life`. In our approach, the membership of an account to a list is viewed as an endorsement from the list creator to that account. In addition, the title of the list and the text describing the list can provide *labels* that describe the topic of endorsement.

The use of Twitter lists for finding topical experts is also exploited by the state-of-the-art method, Cognos [4], but in a different manner. In our approach, we use Twitter lists to build a *directed edge-labeled graph* $G = (V, E, \ell)$, where each edge $e = (u, v) \in E$ represents an endorsement and $\ell(e) \subseteq L$ is a set of labels that describe the topic(s) of the endorsement. The graph G is called *endorsement graph*. To solve the problem of finding topical authorities for a given topic (query) $q \subseteq L$ we provide an adaptation of the personalized PageRank algorithm, where the random walk takes place on the endorsement graph and it is guided by the similarity between the user query and the edge labels.

We complement our link-analysis algorithm with a *focused crawling* method, called BackwardForward, which retrieves a high-quality set of candidate accounts for a given topic, and provides the underlying edge-labeled graph in which to find topical authorities. This focused-crawling algorithm improves tremendously the scalability of our method, as it allows to search for experts on a small subgraph of the whole network, without significant sacrifice in accuracy. Notably, our endorsement graph contains fewer than 0.05% of all Twitter accounts, while in most cases it contains more than 50% of Twitter's "who to follow" (wtf [6]) top-10 results.

Our results show that our method has higher average precision than Cognos for 70% of the queries on both a user study conducted with expert judges as well as on an extensive evaluation on Crowdflower.[1] Our approach is competitive with WTF, outperforming it in the expert user study 70% of times, while being outperformed in the crowd-sourced one. However, our method returns more *serendipitous* results than WTF (54% of the results by our method are judged non-obvious, compared to only 40% by WTF). Overall, the proposed method returns a good balance of well-known and serendipitous results, while being consistently more relevant than COGNOS, and comparable to WTF. We call our algorithm FAME, as we observe that in practice it Finds a mix of Authorities, Mavens, and Experts.

The main contributions of this paper are as follows:

- we introduce FAME, a new method for finding topical authorities on Twitter for a given query topic;
- we formalize the problem as finding important nodes in an edge-labeled graph with respect to a given query;
- we show how to scale FAME with the help of a focused-crawling algorithm BACKWARDFORWARD, which allows to drastically reduce the input graph without losing accuracy;
- we perform an extensive experimental evaluation, using exploratory data analysis, a user study with expert judges in our lab, and a larger-scale user study on Crowdflower. FAME outperforms the current state-of-the-art, COGNOS, in terms of relevance, and is competitive with Twitter's WTF, while providing more serendipitous results.

II. RELATED WORK

Weng et al. [18] propose a variant of PageRank called TWITTERRANK that uses the Twitter follower graph and processes content from tweets to identify topical authorities. Pal and Counts [14] propose a number of features of authors (e.g., number of mentions, number of retweets) to measure author impact and use probabilistic clustering to identify top authors for a given topic. Apart from research studies, Twitter has its own official service called Who-To-Follow (WTF) to find experts for a given topic. WTF integrates a multitude of user and behavior features to generate recommendations [6].

These methods rely on application-specific features, which limit their relevance to a narrow domain and a particular application. On the other hand, FAME is a general-purpose approach that can be applied to any social network with endorsement features. The suitability of the chosen features to identify expertise of a user account is debatable. For instance, identifying influential Twitter users based on the "follow" relationship is prone to errors, since the relationship does not carry a strong indication of influence [18]. Moreover, due to short text and noisy data on microblogs, it is hard to identify the topic that the endorsement refers to. Furthermore, self-reported biographies, which are the most commonly-used signal for finding authorities, can be easily manipulated to fake expertise or for meagre humor. Finally, these methods rely on access to user information of entire Twitter network, which makes the algorithms impractical for anyone but Twitter.

The most recent state-of-the-art work in the field is COGNOS by Ghosh et al. [4]. For each user COGNOS creates a topic vector with expertise labels extracted from Twitter lists, computes cover density ranking between query and topic vectors, and finally returns the ranked results. While our method, similarly to COGNOS, uses Twitter lists as a crowd-sourced signal of *user expertise*, it differs in important ways.

[1]https://www.crowdflower.com

First, COGNOS relies only on the inferred topic vectors, and does not make use of the global link-structure of the graph. By depending solely on topic vectors, they ignore hub and authority information of user accounts which is a key signal in finding experts. Second, it uses a heuristic to compute a topical similarity score between the topic vector for a user and the given query vector. Our algorithm can be viewed as a more principled approach to the same problem. The problem of finding topical authorities is abstracted as the problem of ranking nodes in an edge-labeled graph with respect to a given query. We then devise a query-dependent personalised PageRank for edge-labeled graphs. While this paper focuses on Twitter, the method can be used to find topical authorities in any labeled endorsement graph.

Last but not least, our method is computationally practical in near real-time scenarios, and does not require access to the full Twitter network. The rate limits imposed by Twitter API and the ever changing dynamics of micro-blogging sites pose serious scalability and practicality issues to all the methods which rely on the entire Twitter dataset. Previous works rely on either working with a limited time period dataset or on special access accounts ("firehose" and white-listed machines) to Twitter that have higher or no rate limits. This is the first study which explores alternative options to solve the problem by using focused crawl.

FAME uses a simple approach for focused crawling called BACKWARDFORWARD, which crawls a subset of the entire network, focusing on topical authorities and hubs. As a result of this focused crawl, we obtain a small (less than 0.05% of the entire Twitter network) yet high-quality subgraph. To the best of our knowledge, although focused crawl is a well-studied topic [1, 2, 3, 8, 11], it has not been explored in the context of identifying topical experts on social networks. Since focused crawling is a large area in itself, and orthogonal to the core contribution of this paper, we do not present a comparison of BACKWARDFORWARD with other baselines. However, we present anecdotal results in section VI-B to assess the quality of nodes discovered by the focused crawl.

Historically, there have been many attempts to use global link-structure to identify the importance of a user. The two best-known algorithms that exploit link structure to find authorities are PageRank [13] and HITS [9]. However, they are known for topical drift [17]. Various variants of PageRank and HITS have been proposed to bias the algorithm towards pages containing query words (topic)[7, 12]. The most similar work to FAME in this aspect is QD-PAGERANK by Richardson and Domingos [17]. We highlight the issues in QD-PAGERANK and extend it to address them. We then formulate the problem of finding topical authorities in social networks as a link-analysis problem, and present a query-dependent personalised PageRank approach to find topical experts on Twitter.

III. Data model and preliminaries

As discussed in the introduction, our algorithm for finding topical authorities leverages *Twitter lists*, which are user-curated sets of Twitter accounts relevant to a topic. Consider that a user A creates a list t on a certain topic and adds account B to the list. Furthermore, assume that A uses a set of keywords K to describe the list t. The set of keywords K is extracted by text-processing techniques from the title and the textual description of the list. We can then interpret the membership of account B in the list of user A as an endorsement, and model it as an edge $A \to B$ in a graph that contains all Twitter accounts. In addition, we use the set of keywords K to label the edge $A \to B$.

The resulting data model is a directed edge-labeled graph $G = (V, E, \ell)$, where V represents the set of all Twitter accounts, and edges represent endorsements extracted from lists. We call G the *endorsement graph*.

We assume that L is the set of all possible labels, and $\ell(e) \subseteq L$ is the set of labels associated with edge $e \in E$. Furthermore, when seeking for an authority with respect to a given topic, we assume that the topic of interest can be expressed as a query in the form of a subset of the same ground set of labels L, i.e., $q \subseteq L$. The problem of finding topical authorities is then abstracted as a problem of ranking nodes in an edge-labeled graph with respect to a given query.

*Problem 1 (*Topical Authorities): Given a directed edge-labeled graph $G = (V, E, \ell)$, with $\ell : E \to 2^L$, where L is a ground set of labels, and given a query $q \subseteq L$, find a ranking of the nodes of V with respect to the query q.

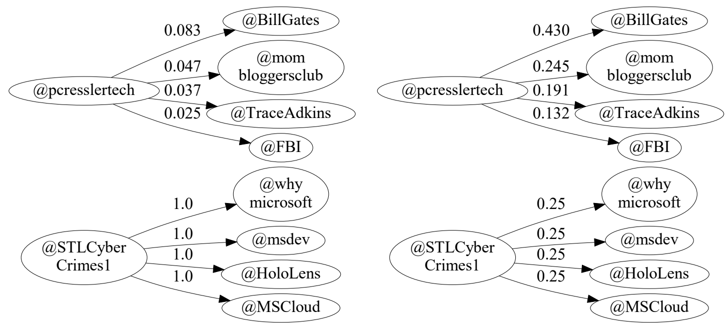
We assume that the weight on the edge $e = i \to j$ is $w_q(i \to j) \equiv w_q(e)$ given by the similarity between the query q and an edge label set $\ell(e)$ measured by a function $\mathrm{sim}(q, \ell(e))$. We wish that the value of the similarity function can be non-zero even when $q \cap \ell(e) = \varnothing$. For example, if $q = \{\texttt{`soccer'}\}$ and $\ell(e) = \{\texttt{`football'}\}$ it is fairly obvious that a non-zero value of $\mathrm{sim}(q, \ell(e))$ is desirable. As discussed in the next section, we explore different options for defining $\mathrm{sim}(q, \ell(e))$. In particular we define $w_q(e) = \mathrm{sim}(q, \ell(e))$ over a *label space*, a *people space*, or by using a *word-to-vec model* [10].

IV. Algorithms

Our algorithm uses ideas by Richardson and Domingos [17], who propose a more intelligent surfer called QD-PageRank (*query-dependent* PageRank). For a node i, let F_i be the set of nodes that i points to (forward nodes), and B_i the set of nodes that point to i (backward nodes). Furthermore, assume that, given a query q, a relevance score $R_q(j)$ can be computed for each graph node j by using $w_q(i \to j)$ for all $i \in B_j$. The node relevance scores $R_q(j)$ are normalized to form a probability distribution $P'_q(j)$, for all graph nodes j. The edge weights $w_q(i \to j)$ are also normalized to probabilities $P_q(i \to j)$.

More in detail,

$$P'_q(j) = \frac{R_q(j)}{\sum\limits_{k \in V} R_q(k)}, \text{ and } P_q(i \to j) = \frac{w_q(i \to j)}{\sum\limits_{k \in F_i} w_q(i \to k)}.$$



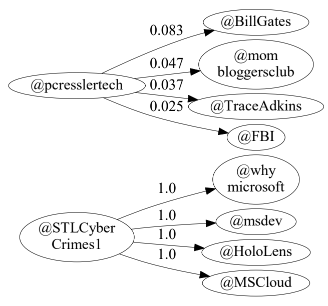
(a) ($w_q(e) = \mathrm{sim}(q, \ell(e))$) **(b)** (QD-PageRank)

Fig. 1: Effect of stochastic normalization

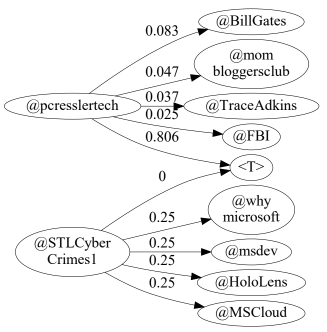
Given query q, the QD-PageRank value for each node in the graph is determined by the stationary distribution of a random walk that has edge transition probabilities given by P_q and teleportation probabilities given by P'_q. When a node's out-links have all zero relevance, or when a node has no out-links, the surfer teleports by choosing a new page according to P'_q.

Effects of stochastic normalization. The normalization step of the QD-PageRank algorithm, as described above, has an undesirable effect. In particular, as QD-PageRank normalizes the node-relevance scores $R_q(j)$ and edge-weights $w_q(i \to j)$ to obtain a row-stochastic transition matrix, the entries of the row-normalized transition matrix represent only the *relative importance* of the nodes in the same row in the stochastic matrix, rather than an absolute one. To illustrate this phenomenon, Figure 1a depicts two nodes from the Twitter graph, which have the same out-degree. One node has out-links with high relevance to the query "Microsoft" whereas the other one points to low-relevance accounts. The result of stochastic normalization can be seen in Figure 1b. The node `@mombloggersclub`, which initially has edge-weight $w_q(e)$ = 0.047, after normalization gets considerably boosted to 0.245. While the normalized edge weights are proportional to the initial weights, the normalization considers only local information. On the global graph, `@mombloggersclub` has now similar weight to `@whymicrosoft` and `@msdev`. Thus, in essence, the stochastic normalization loses some of the information contained in the initial weights.

Proposed approach. To address such undesirable normalization effects, we propose an improved query-dependent PageRank algorithm, which makes use of the *teleportation vector*. The proposed algorithm, named PREP (for PageRank on Endorsement graPh), given a query q, creates a personalized teleportation vector T_q of size $|V|$, where the weight for each node $v \in V$ is proportional to a similarity score $\mathrm{sim}(q, v)$ (details follow). For a teleportation coefficient α, the PageRank is defined as follows: when the random walker is at a node i it follows an out-link $j \in F_i$ with probability $(1-\alpha)\frac{\gamma}{\beta}\, w_q(i \to j)$. With probability $\alpha + (1-\alpha)(1-\gamma)$ it teleports to one of the nodes in T_q (following the probability distribution in T_q) where $\beta = \sum_{j \in F_i} w_q(i \to j)$ and $\gamma = \min\{1, \beta\}$.



(a) $(w_q(e) = \mathrm{sim}(q, \ell(e)))$



(b) (PREP)

Fig. 2: Modified personalised PageRank algorithm.

The effect of this process is that when the outgoing edge probabilities $w_q(i \to j)$ are all small, and $\beta \ll 1$, then $1-\gamma \approx 1$, so the probability of following a graph edge decreases and the probability of a random jump increases. On the other hand, when $\beta \geq 1$, the probability of a random jump is just α.

Considering the same example again in Figure 2, now with the new weighting ($\alpha = 0$), we can observe that the edge weight of `@mombloggersclub` remains 0.047, and the probability of a random jump to a node in the teleportation vector increases.

It remains to specify how we compute the personalized teleportation vector T_q and the weight of each edge (i.e., the similarity $\mathrm{sim}(q, \ell(e))$ of the query q to the set of labels associated to the edge $\ell(e)$).

Teleportation vector. For each node $j \in V$ we consider a vector $\mathbf{v}_j$ over the space of labels, where the coordinate $\mathbf{v}_j(x)$ counts the number of times that node j has been endorsed with label x by all other nodes in the graph G. Then the personalized teleportation vector T_q is computed so that $T_q(j)$ is proportional to the cosine similarity $\cos(q, \mathbf{v}_j)$.

Similarity function. There are several possible ways to compute $\mathrm{sim}(q, \ell(e))$. We explore several "spaces" where to define this similarity function.

1. **Label space:** We represent q and $\ell(e)$ by indicator vectors $\mathbf{w}_L(q)$ and $\mathbf{w}_L(\ell(e))$ in a vector-space of dimension $|L|$. The similarity $\mathrm{sim}(q, \ell(e))$ is defined as the cosine of the corresponding vectors, that is, $\mathrm{sim}(q, \ell(e)) = \cos(\mathbf{w}_L(q), \mathbf{w}_L(\ell(e)))$. This function captures only syntactic similarity.
2. **People space:** We represent each label in q and $\ell(e)$ by indicator vectors over the people who have been endorsed for that label. We then compute the similarity between two labels as the cosine between the corresponding vectors. Finally, we aggregate over the set of labels in q and $\ell(e)$ by taking the maximum. Let $\mathbf{w}_P(x)$ represent the set of nodes that have an incoming edge with label x. We define
$$\mathrm{sim}(q, \ell(e)) = \max_{x \in q, y \in \ell(e)} \cos(\mathbf{w}_P(x), \mathbf{w}_P(y)). \tag{1}$$
This similarity captures the co-references by different Twitter users who may refer to the same entity in similar yet syntactically different ways. For instance, a user is using the label `'football'` and another is using `'soccer'` to describe players of Manchester United.
3. **Word2vec space:** We represent q and $\ell(e)$ as vectors in a high-dimensional space. The space is defined by a *word embedding* algorithm, word2vec [10]. The similarity is then computed as the cosine similarity between the vectors corresponding to the labels in the embedding. Let $\mathbf{w}(x)$ denote the vector representation of label x, and $\mathbf{w}(q) = \sum_{x \in q} \mathbf{w}(x)$ and $\mathbf{w}(\ell(e)) = \sum_{x \in \ell(e)} \mathbf{w}(x)$. We define
$$\mathrm{sim}(q, \ell(e)) = \cos(\mathbf{w}(q), \mathbf{w}(\ell(e))). \tag{2}$$
It has been conjectured that this kind of word embeddings actually captures similarity at a semantic level. Given the high sparsity of our data, we rely on a pre-trained model of word vectors trained on about 100 billion words from Google News by using Gensim [16].

V. BUILDING THE ENDORSEMENT GRAPH

This section describes how to create the edge-labeled graph (endorsement graph) used as input for our algorithm.

Focused-crawling algorithm. Exhaustive crawling of the entire Twitter graph, which consists of approximately 300 million active users, is practically impossible given the rate-limit on the Twitter API [4]. Further, query dependent PageRank approaches require considerable processing time and storage [15]. Storing the entire Twitter graph and processing it in online for each query will be a hindrance for real-time constraints. To overcome these challenges we propose a novel focused crawling approach called BACKWARDFORWARD.

We start with a set of seed nodes on a broad topic. This seed set can be manually specified as background knowledge, or can be bootstrapped from other search systems. These seeds act as the initial set of hubs. Our crawling is an iterative process in which we use Twitter list-membership information to extract endorsement. As discussed earlier, the membership of an account to a list is viewed as an endorsement from the list owner to that account. In each iteration, we alternately perform a forward step to find authorities (nodes which are endorsed by hubs) and a backward step to find hubs (nodes which endorse authorities). At the end of each iteration we prune the graph by retaining only the top-k hubs and authorities by ranking in out-degree and in-degree, respectively. This pruning of search space allows for the crawl to grow while making sure that only promising nodes(good authorities and good hubs) are added to the graph. Given that we are only interested in finding experts, pruning nodes with low authority and hub score has limited impact. This reduced space allows to find most of the relevant nodes for a given topic without exploring the full social network.

Extracting *expertise* meta-data from Twitter lists. Each list has an owner, name, description, and members. We create a directed edge from the owner of the list to each of its members. We then process list name and description to infer the label set of the endorsement for each edge. We extract syntactic tokens

TABLE I: Most frequent labels extracted from the Twitter lists for some well-known users.

User	Most frequent labels
Barack Obama	politics, news, world, left, government, leaders, international, president
Bill Gates	business, technology, innovation, microsoft, rich, thought leaders, medicine, giving
Conan O'Brien	comedy, entertainment, news, funny, humor, laugh
Edward Snowden	news, politics, security, internet, activist, celebritis, journalists, rights, privacy, technology
Dalai Lama	religion, chinese, popular users, culture, self, help, quotes, wisdom, health, nonprofit causes

TABLE II: Average number of followers and most common labels extracted from the name and description of the top-20 authorities in each iteration of BackwardForward.

Iter.	**Avg # Followers**	**Most common labels extracted**
1	232 791	rugby, player, irish, ff, ambassador, rugby player, leicester tigers, professional, barbarian ff, ff rugby
5	371 450	rugby, sky sports football, england, union, football, super, player, young, black, england rugby
10	243 480	rugby, young, suite, us, japan, england rugby, bc, athlete, ff rugby, leicester tigers
20	942 598	rugby, football, league, route, player, club, premier, home, premier league, sky sports football
40	932 372	rugby, football, super, player, premier league, home, young, sky sports football, back, oval ball
60	936 307	rugby, football, union, ff rugby, ebb rugby, oval ball, twitter rugby football union, rugby player, super, club

using standard tokenizers. Since many of list descriptions and names are written in CamelCase (e.g., "MachineLearning") we leverage this knowledge while tokenizing by splitting CamelCased words. We then apply common natural-language processing techniques such as case-folding, stop-word removal, and stemming. In addition since names of topics can be phrases such as "English premier league" and entities such as "Microsoft," we extract noun phrases and named entities.

The final result is a set of labels for each edge. Examples of labels extracted from Twitter lists to describe some popular accounts are given in Table I.

VI. Experimental results

In this section we evaluate the proposed algorithms. Among several evaluation experiments, we also report the results of two user studies (Section VI-D), a small-scale one with expert judges, and a large-scale crowd-sourced one on Crowdflower.

A. Dataset

In order to build our endorsement graph we perform a focused crawl with $k = 200$ and use as seeds a collection of 854 international Rugby Union players, clubs, and organizations, 248 English Premier League football players and clubs [5], and Twitter handles of 6 major computer science conferences. In total we process 37 183 lists and create an edge-labeled endorsement graph with 139 798 nodes, 190 435 directed edges, and 17 887 unique labels.

B. Quality of nodes discovered

(i) We use the number of followers of a user, and labels extracted from the Twitter name and description, as a proxy to user authoritativeness and relevance to the topic. Table II shows these values for top-20 authorities discovered in different iterations. We observe that the quality of nodes discovered by BackwardForward is high, and improves (average degree increases steadily), while the nodes are relevant to the topic. In addition to identifying the most common label '`rugby`', the algorithm extracts other relevant labels (e.g., '`oval ball`').

(ii) Next we compare the extent to which the experts identied by official Twitter Who-To-Follow service (WTF) can be recalled in our dataset (WTF has access to a much wider pool of accounts to draw from). We select the 10 most common labels in our dataset and use these as queries for Twitter's search engine (WTF). For each query, we compute the fraction of top 10 results provided by Twitter that can be recalled in our dataset. Table III shows that for 9 out of 10 topics, more than half of top 10 results provided by Twitter appear in our crawl.

TABLE III: Fraction of top-10 WTF results that appear in the collected dataset. The dataset contains 140 k accounts out of ≈ 300 million active accounts on Twitter (fewer than 0.05%).

Query	Fraction of top 10 results	Query	Fraction
open source	0.8	entrepreneur	0.6
programming	0.7	football news	0.5
manchester united	0.7	inspiring women	0.5
charity	0.7	microsoft	0.8
cloud experts	0.1	robotics	0.6

TABLE IV: Selected results provided by FAME.

Query	Top results
Comedy	Jerry Seinfeld, Chris Rock, Jim Carrey
Machine Learning	ICML Conference, Deep Mind
Business News	Forbes, Warren Buffet
Manchester United	Wayne Rooney, David De Gea
Entertainment	The X files, Last Week Tonight, The Daily Show
Poker	Red Chip Poker , Bryn Kenney, Poker Dealer
Inspiring Women	Lean In, Systers Community, She++
Programming	Martin Odersky, Codecademy, Girls Who Code

C. Evaluating the expert-finding algorithm

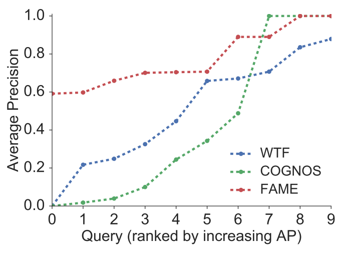
An important component of PREP is the similarity function between query and edge, which induces the weights of the endorsement graph. We evaluate the spaces used to define the similarity function, presented in Section IV, and select the best performing one.

Word2vec space. Due to the sparsity of labels in our dataset, we use an embedding from a model trained on about 300 billion words from a Google News dataset. However, the similarity scores obtained by word2vec are not suitable for our purpose. As such, we do not investigate this version any further, and report some anecdotal result: Words such as "sports" and "players" are as similar as "sports" and "politics" (cosine similarity of 0.3). In addition, the model lacks much of the domain-specific vocabulary present in our labels. For instance, "football" and "arsenal" have a cosine similarity of -0.01. As a result, the PREP ranking with word2vec similarity tends to just highlight popular accounts.

People space. In "people space" we represent labels as a set of people who have been endorsed for the specific label.

These vectors are then used to compute cosine similarity. To assess the quality of the similarity function, we randomly select 30 000 pairs of labels from the dataset, and give them as input queries to FAME. For each pair of queries, we compute the correlation between the resulting rankings. Ideally, we would like similar queries in people space to return similar rankings. We find that this is the case indeed: the Pearson correlation between query similarity and ranking similarity (measured with Kendall-τ rank correlation) is 0.83. Unfortunately, even though the results of this similarity function are quite desirable, the high sparsity of the resulting similarity matrix restricts its use. Indeed, most pairs of labels in our dataset have zero similarity in people space (98.6%). Since the main purpose of people-space approach is to derive smooth similarity score, we end up not gaining much at the expense of increased online computation.

Label space. In "label space" we define a similarity function as the cosine similarity between edge labels. In our experiments "label space" performed better than the other two spaces. Our algorithm, FAME, uses cosine similarity in this "label space." Table IV lists selected results provided by FAME for a small set of queries. Interestingly, the results from FAME are quite diverse. Some of the results are large organizations (Forbes), famous personalities (Warren Buffet), and popular channels (Last Week Tonight, The Daily Show). At the same time, FAME also finds smaller but interesting channels (Girls Who Code, Poker Dealer), as well as domain specific authorities (ICML Conference, Red Chip Poker, Codecademy), and highly relevant individual users (Martin Odersky, Bryn Kenney). Since FAME does not use measures such as follower counts or any other signal of popularity, it provides small organizations and individual users an equal opportunity to appear in the results. Thus, the final ranking is an interesting mix of results that contains obvious as well as serendipitous results. We return to this point of "serendipity" and evaluate it empirically in our user study. In our experimental evaluation, we also compare QD-PAGERANK with FAME. As expected from our initial observations, FAME outperforms QD-PAGERANK. Due to the lack of space we omit these results, and we discuss comparison of FAME with stronger baselines.


(a) AP(τ_1)

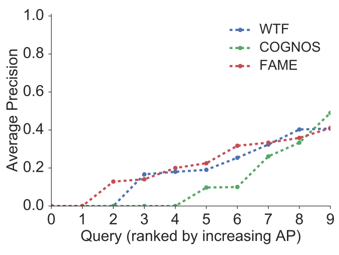
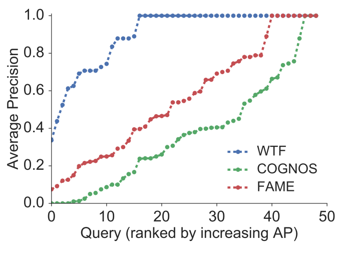

(b) AP(τ_2)

Fig. 3: User Study 1 (expert): Average precision for top 10 results for the 3 ranking methods across 10 queries. Queries on the x-axis are ranked in increasing order of average precision for each method.


(a) AP(τ_1)

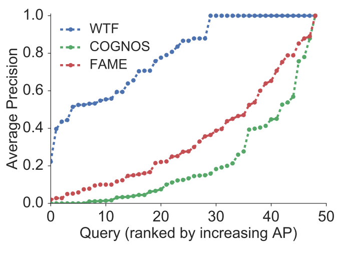

(b) AP(τ_2)

Fig. 4: User Study 2 (crowdsourced): Average precision for top 10 results for the 3 ranking methods across 49 queries. Queries on the x-axis are ranked in increasing order of average precision for each method.

TABLE V: User Studies: Mean average precision across queries for the 3 ranking methods. Values marked by an asterisk (*) indicate statistically significant differences between methods (all apart from User Study 1 (expert) at threshold τ_2).

(a) User Study 1 (expert)

Method	MAP(τ_1)	MAP(τ_2)
FAME	0.774*	0.211
COGNOS	0.423*	0.128
WTF	0.499*	0.192

(b) User Study 2 (crowdsourced)

Method	MAP(τ_1)	MAP(τ_2)
FAME	0.423*	0.280*
COGNOS	0.275*	0.177*
WTF	0.816*	0.754*

TABLE VI: Comparison of average precision across queries. Values in the table are fraction of queries for which the said ranking method has higher average precision than the other.

Ranking method	User study 1	User study 2
FAME $\succ$ COGNOS	70%	67%
FAME $\succ$ WTF	70%	6%
COGNOS $\succ$ WTF	40%	2%

D. User study

We compare FAME to the state-of-the-art system COGNOS [4] and the official Twitter Who-To-Follow (WTF) service.

Setting. First, we perform a smaller expert user study on 10 queries (selected uniformly at random from the label set), with 7 expert judges per result. Results from this user study with high agreement are then used as "gold" set for quality control in the larger-scale evaluation on Crowdflower. The final experimental design on Crowdflower has 49 queries (selected uniformly at random from the label set.), with 5 judges per result. In both studies, for each query we randomly mix the top 10 results from FAME with 10 from Twitter's WTF and COGNOS each. We then give them to the judges who are asked to rate each result on a 3-point scale — "Not interested"(0), "Somewhat interested"(1), and "Very interested"(2). To quantify the "serendipity" of each relevant account, we ask the evaluators to evaluate the results on "How expected the result is" on a 2-point scale — "Obvious, I already expected to know about it," or "Interesting, I would not have thought about it." With this second question we aim at quantifying how conservative a ranking method is. That is, whether it gives importance to well-known, safe, but obvious

TABLE VII: Serendipity: Comparison of obvious vs. interesting results across all topics for the 3 ranking methods.

Results	WTF	FAME	COGNOS
Obvious	59.9%	45.6%	40.1%
Interesting	40.1%	54.4%	59.9%

TABLE VIII: Examples of a) results of WTF unanimously judged NOT relevant in the user study and b) results of FAME unanimously judged as relevant in the user study

Query	User	Twitter name	Twitter bio
(a) Representative example for which name or bio contain query terms.			
programming	@PIUpdate	Programming Insider	Programming Insider is your proven daily source for everything media.Ratings, trends, observations, breaking news, scheduling, trivia and much more!
cloud experts	@CloudExperts	Nitesh Pandey	Office365 reseller, Sales trainer , Fan of Test Cricket, Devout Atheist! say Hi!
(b) Representative examples for which name and bio do not contain query terms.			
microsoft	@BillGates	Bill Gates	Sharing things I'm learning through my foundation work and other interests.
programming	@Android	Android	News, tips, and tricks direct from the Android team.

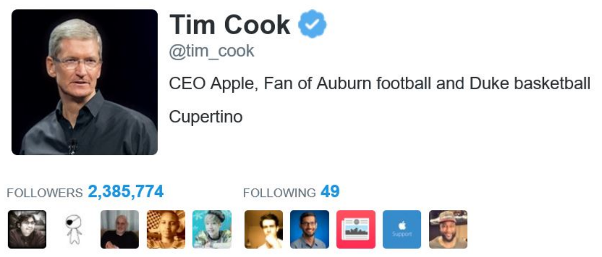
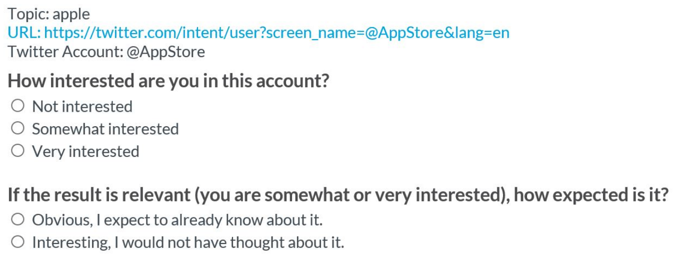

Tim Cook
@tim_cook
CEO Apple, Fan of Auburn football and Duke basketball
Cupertino
FOLLOWERS 2,385,774 FOLLOWING 49

Fig. 5: Survey screen which shows a user's Twitter name, handle, bio information, and follower details.


Topic: apple
URL: https://twitter.com/intent/user?screen_name=@AppStore&lang=en
Twitter Account: @AppStore
How interested are you in this account?
○ Not interested
○ Somewhat interested
○ Very interested

If the result is relevant (you are somewhat or very interested), how expected is it?
○ Obvious, I expect to already know about it.
○ Interesting, I would not have thought about it.

Fig. 6: Survey screen for each result, asking the evaluators to judge relevance of a result and its serendipity (obvious or interesting).

results over interesting yet relevant ones that can be slightly risky. Figures 5 and 6 show two screenshots of the survey.

Results. We aggregate the judgements from all the evaluators by taking the median value for each result. We discard judgements for which the confidence (a weighted agreement on the judgement) is below 0.5. Then, we apply two different thresholds to binarize the results: in the first case, we consider relevant those results that receive a score of at least '1' ("somewhat interested"), in the second case, only those with score '2' ("very interested"). We refer to the first case as 'τ_1' and to the second as 'τ_2.' We use *average precision* (AP) and *mean average precision* (MAP) as relevance metrics, and compute one value for each of the two thresholds. All values reported in the tables (apart from User Study 1 at threshold τ_2) have shown to be statistically significant in a dependent t-test of pairwise comparison of the systems (FAME vs. COGNOS and FAME vs. WTF) on *average precision* (AP) for all queries.

Before we look at experimental results, it is worth mentioning that the underlying endorsement graph for FAME contains less than 0.05% of the entire Twitter network, whereas COGNOS and WTF have access to access to the full Twitter network. Further, FAME's underlying graph is only constructed for topics "rugby", "football", and "computer science", however many of the randomly selected queries used in the user study are not related to these topics. For such queries FAME naturally has limited set of nodes to select from, whereas COGNOS and WTF have access to the full Twitter network.

The experimental results for the expert user study are shown in Figure 3. The plot shows the scores for all the queries, ranked by increasing AP. The left plot shows the average precision (AP) scores for τ_1, while the right one for τ_2. This visualization allows to easily compare the three ranking methods on their best and worst performing queries. Figure 4 reports the same measures for the crowdsourced user study.

Figures 3 and 4 show that FAME consistently performs better than COGNOS for both thresholds. In addition, COGNOS returns no relevant result for 40% of queries in the first study, and 20% of queries in the second one. Conversely, FAME only returns irrelevant results in a single case in the first study. Recall from Table III that we have already noticed that our crawl does not contain results that match Twitter's one for one of the queries ("cloud experts"). This fact is now reflected in the AP scores.

Table V shows MAP computed across all queries for each user study. FAME performs better than COGNOS in both the user studies, whereas WTF outperform FAME in user study 2.

Table VI provides a query wise comparison of the three systems. FAME performs better than COGNOS on 70% of the queries, showing clear distinction in the performance of the two algorithms in spite of the fact that both use Twitter lists as input. In case of FAME vs. WTF though, the results are not as unanimous. While in the expert user study FAME has higher AP than WTF for 70% of the queries, the results are very different in the crowdsourced user study (only 6%). A similar effect can be seen for COGNOS, as its advantage falls from 40% to 2%. We believe that, in large part, this result is influenced by the conditions of the evaluation. Platforms such as Crowdflower are based on a reward system for accuracy of evaluators. Accuracy is measured by using a "gold" set of questions, and wrong answers on gold questions are penalized. As a consequence, evaluators on these platforms tend to be more conservative, and thus wary of voting less known accounts as relevant.

In order to realistically quantify this effect, we compare

FAME with other methods on serendipity, by looking at the second question in the study. Once again we aggregate the judgements and retain only those with higher than 50% inter-annotator agreement. Table VII shows that WTF has the highest number of "obvious" results and the lowest number of "interesting" ones. It is clear that WTF prefers popular and "safe" results and leaves little room for exploration to show relevant but "risky" results. Both FAME and COGNOS show a majority of interesting results, among the relevant ones. Taking into consideration also the relevance as discussed above, FAME strikes a good balance between WTF (which favors relevance and safety) and COGNOS (which favors serendipity).

Potential for Twitter lists. Our study suggests that WTF heavily relies on user's self reported content, such as screen name, handle, and bio information of the user. While these signals can be accurate, relying heavily on self-reported content allows to misuse the feature for self-promotion and advertising, or even for malicious purposes. One can easily boost their ranking in the results by using appropriate keywords. Table VIII gives examples of some WTF results that are unanimously judged as "not relevant" in the user study. At the same time, high reliance on self-reported content prevents relevant and interesting users from appearing in the result. The table also gives example results by FAME which are unanimously judged as relevant and interesting. None of these accounts contain the query in their bio-information. As expected, none of these appear in the top-10 WTF results. In addition the results from WTF mostly include organizations and popular personalities whereas FAME results have a fair mix of large organizations, personalities, personal accounts, and smaller business accounts. We conclude that using Twitter lists can be a reliable source of crowd-sourced expertise meta-data, and can be used as a powerful supplementary signal in identifying topical authorities.

VII. CONCLUSIONS

In this paper we presented FAME, a novel approach to find topical authorities on Twitter. We showed how to use Twitter Lists to create an elegant problem formulation based on an edge-labeled directed weighted graph called an *endorsement graph*. Given such a graph, the problem of finding topical authorities can be modeled as finding important nodes in a graph appropriately and dynamically weighted according to the input query. To solve this latter task, FAME employs a variant of query-dependent personalized PageRank customized to the problem. We also presented a focused crawling strategy called BACKWARDFORWARD that enables to collect a high-quality graph by exploring just a fraction of the whole network.

Given a query topic, FAME is able to find experts that not only are relevant, but also serendipitous, i.e., it pushes the boundaries by returning users who are not famous, and thus, obvious. Our extensive experimental evaluation showed that FAME significantly improves on QD-PAGERANK, and outperforms COGNOS, the state-of-the-art in research. In addition, FAME is competitive with Twitter's WTF while giving more unexpected results, even though it has access to just a fraction of the data available to WTF.

Limitations. Currently FAME is limited to syntactic matching rather than semantic one. While using the word2vec or people space would take care of this limitation, our experimental results showed that the simple label space outperforms them. Our word2vec model relied on a pre-trained model, but given enough data training a model for the specific task should be feasible, and could possibly yield better results. Second, our dataset is crawled from a specific set of seeds. The effect of this choice on the ranking algorithm needs further investigation.

Acknowledgements. This work was supported by the Tekes project "Re:Know," the Academy of Finland project "Nestor" (286211), and the EC H2020 project "SoBigData" (654024).

VIII. REFERENCES

[1] C. C. Aggarwal, F. Al-Garawi, and P. S. Yu. Intelligent crawling on the world wide web with arbitrary predicates. In *WWW*, pages 96–105. ACM, 2001.

[2] S. Chakrabarti, M. Van den Berg, and B. Dom. Focused crawling: a new approach to topic-specific web resource discovery. *Computer Networks*, 31(11):1623–1640, 1999.

[3] M. Diligenti, F. Coetzee, S. Lawrence, C. L. Giles, M. Gori, et al. Focused crawling using context graphs. In *VLDB*, pages 527–534, 2000.

[4] S. Ghosh, N. Sharma, F. Benevenuto, N. Ganguly, and K. Gummadi. Cognos: crowdsourcing search for topic experts in microblogs. In *SIGIR*, pages 575–590. ACM, 2012.

[5] D. Greene and P. Cunningham. Producing a unified graph representation from multiple social network views. In *WebSci*, pages 118–121. ACM, 2013.

[6] P. Gupta, A. Goel, J. Lin, A. Sharma, D. Wang, and R. Zadeh. WTF: The who to follow service at Twitter. In *WWW*, pages 505–514, 2013.

[7] T. H. Haveliwala. Topic-sensitive PageRank: A context-sensitive ranking algorithm for web search. *TKDE*, 15(4):784–796, 2003.

[8] M. Hersovici, M. Jacovi, Y. S. Maarek, D. Pelleg, M. Shtalhaim, and S. Ur. The shark-search algorithm. an application: tailored web site mapping. *Computer Networks and ISDN Systems*, 30(1):317–326, 1998.

[9] J. M. Kleinberg. Authoritative sources in a hyperlinked environment. *JACM*, 46(5):604–632, 1999.

[10] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *NIPS*, pages 3111–3119, 2013.

[11] S. Mukherjea. Wtms: a system for collecting and analyzing topic-specific web information. *Computer Networks*, 33(1): 457–471, 2000.

[12] L. Nie, B. D. Davison, and X. Qi. Topical link analysis for web search. In *SIGIR*, pages 91–98. ACM, 2006.

[13] L. Page, S. Brin, R. Motwani, and T. Winograd. The PageRank citation ranking: bringing order to the Web. 1999.

[14] A. Pal and S. Counts. Identifying topical authorities in microblogs. In *WSDM*, pages 45–54. ACM, 2011.

[15] F. Qiu and J. Cho. Automatic identification of user interest for personalized search. In *WWW*, pages 727–736. ACM, 2006.

[16] R. Řehůřek and P. Sojka. Software Framework for Topic Modelling with Large Corpora. In *NLPFrameworks*, pages 45–50, 2010.

[17] M. Richardson and P. Domingos. The intelligent surfer: Probabilistic combination of link and content information in PageRank. *NIPS*, 2:1441–1448, 2002.

[18] J. Weng, E.-P. Lim, J. Jiang, and Q. He. Twitterrank: Finding Topic-Sensitive Influential Twitterers. In *WSDM*, pages 261–270. ACM, 2010.