On Local Expert Discovery via Geo-Located Crowds, Queries, and Candidates

WEI NIU, ZHIJIAO LIU, and JAMES CAVERLEE, Texas A&M University

Local experts are critical for many location-sensitive information needs, and yet there is a research gap in our understanding of the factors impacting who is recognized as a local expert and in methods for discovering local experts. Hence, in this article, we explore a geo-spatial learning-to-rank framework for identifying local experts. Three of the key features of the proposed approach are: (i) a learning-based framework for integrating multiple user-based, content-based, list-based, and crowd-based factors impacting local expertise that leverages the fine-grained GPS coordinates of millions of social media users; (ii) a location-sensitive random walk that propagates crowd knowledge of a candidate's expertise; and (iii) a comprehensive controlled study over AMT-labeled local experts on eight topics and in four cities. We find significant improvements of local expert finding versus two state-of-the-art alternatives, as well as evidence for the generalizability of local expert ranking models to new topics and new locations.

CCS Concepts: • **Information systems** → **Learning to rank**; **Expert search**; *Data stream mining;*

Additional Key Words and Phrases: Local expert discovery, Twitter list, geo-spatial

ACM Reference Format:
Wei Niu, Zhijiao Liu, and James Caverlee. 2016. On local expert discovery via geo-located crowds, queries, and candidates. ACM Trans. Spatial Algorithms Syst. 2, 4, Article 14 (November 2016), 24 pages.
DOI: http://dx.doi.org/10.1145/2994599

1. INTRODUCTION

Identifying *experts* is a critical component for many important tasks, including search and recommendation systems, question-answer platforms, social media ranking, and enterprise knowledge discovery. Indeed, there has been a sustained research effort to develop algorithms and frameworks to uncover experts; for example, Balog et al. [2006], Campbell et al. [2003], Chi [2012], Ghosh et al. [2012], Liu et al. [2005], Pal and Counts [2011], Weng et al. [2010], and Zhang et al. [2007a, 2007b]. These efforts have typically sought to identify *general topic experts*—like the best Java programmer on github—often by mining information-sharing platforms like blogs, email networks, or social media. However, there is a research gap in our understanding of *local experts*. Local experts, in contrast to general topic experts, have specialized knowledge focused around a particular location. To illustrate, consider the following two local experts:

This work was supported in part by NSF grant IIS-1149383 and a Google Research Award. Any opinions, findings and conclusions or recommendations expressed in this material are the author(s) and do not necessarily reflect those of the sponsors.
This article is an expansion and follow-up of Niu et al. [2016], which appeared in the Proceedings of Advances in Information Retrieval: 38th European Conference on IR Research, ECIR 2016, 803-809.
Authors' addresses: W. Niu, Z. Liu, and J. Caverlee, Department of Computer Science and Engineering, Texas A&M University, College Station, TX 77843-3112, USA; emails: {wei, zliu4372, caverlee}@cse.tamu.edu.

DOI: http://dx.doi.org/10.1145/2994599

—A "health and nutrition" local expert in San Francisco is someone who may be knowledgeable about San Francisco-based pharmacies, local health providers, local health insurance options, and markets offering specialized nutritional supplements or restricted diet options (e.g., for gluten allergies or strictly vegan diets).
—A "techie" local expert in Seattle is someone who is knowledgeable about the local tech scene and may be able to answer local information needs like: who are knowledgeable local entrepreneurs, what are the tech-oriented neighborhood hangouts, and who are the top local talent (e.g., do you know any experienced, available web developers?).

Identifying local experts can improve location-based search and recommendation and create the foundation for new crowd-powered systems that connect people to knowledgeable locals. Furthermore, after these local experts have been detected, their knowledge can provide the foundation for many new location-centric information applications. Examples include: (i) local query answering, whereby complex information needs that cannot be satisfied by traditional search engines could be routed to knowledgeable locals; (ii) curated social streams, whereby Facebook and Twitter-like social streams can be reorganized to focus on locally significant events and topics (e.g., to ameliorate panic during disease outbreaks by alerting residents of facts from local health experts rather than on global reports—in the recent panic surrounding Ebola, Dallas residents could be made aware of local precautions rather than focusing on nationwide news summaries); (iii) location-based recommendations, whereby entities of interest to local experts can be recommended to newcomers (e.g., to recommend good venues for meeting local entreprenuers); and on and on. A critical first step in all these cases is in the *identification of local experts*.

Compared to general topic expert finding, there has been little research in uncovering these local experts. Most existing expert finding approaches have typically focused on either small-scale, difficult-to-scale curation of experts (e.g., a magazine's list of the "Top 100 Lawyers in Houston") or on automated methods that can mine large-scale information sharing platforms, such as Balog et al. [2006], Campbell et al. [2003], Chi [2012], Ghosh et al. [2012], Liu et al. [2005], Pal and Counts [2011], Weng et al. [2010], and Zhang et al. [2007a, 2007b]. These approaches, however, have typically focused on finding general topic experts rather than *local experts*. And yet there is growing evidence of the importance of location-centered services: According to a recent Pew Research study: "Location tagging on social media is up: 30% of social media users now tag their posts with their location. For mobile location services, 74% of smartphone owners get directions or other information based on their current location, and 12% use a geo-social service such as Foursquare to 'check in' to locations or share their whereabouts with friends" Zickuhr [2013].

Hence, our focus in this article is on developing robust models of *local expertise* that opportunistically leverage this recent rise of *location* as a central organizing theme of how users engage with online information services and with each other. Concretely, we propose and evaluate a geo-spatial learning-to-rank framework called **LExL** for identifying local experts that leverages the fine-grained GPS coordinates of millions of Twitter users and their relationships in Twitter lists, a form of crowd-sourced knowledge. The framework investigates multiple classes of features that impact local expertise including: (i) user-based features (e.g., the number of users a candidate is following, the number of posts this candidate has made); (ii) tweet content features (e.g., tweet-based entropy of a candidate, the TFIDF score of a topic keyword in a candidate's tweets); (iii) list-based features (e.g., the number of lists the candidate is a member of, the number of lists the candidate has created); (iv) local authority features (e.g., the distance between candidate and the query location, the average distance from a candidate's

labelers to the candidate); and (v) features based on a location-sensitive random walk that propagates crowd knowledge of a candidate's expertise.

Through a controlled study over Amazon Mechanical Turk, we find that the proposed local expert learning approach results in a large and significant improvement in Precision@10, NDCG@10, and in the average quality of local experts discovered versus two state-of-the-art alternatives. We additionally investigate the relative impact of different classes of features and examine the generalizability of the approach in terms of reusing the learned model in different topics. Our findings indicate that careful consideration of the relationships between the location of the query, the location of the crowd, and the locations of expert candidates can lead to powerful indicators of local expertise. We also find that high-quality local expert models can be built with fairly compact features, meaning that these models can be potentially adapted to more constrained scenarios (e.g., in domains with only partial features). Finally, we find that the proposed local expertise models are generalizable: In many scenarios, local experts can be discovered on new topics and in new locations, which is important for uncovering previously unknown experts in emerging areas or in nascent communities.

2. RELATED WORK

Expertise retrieval has long been recognized as a key research challenge [Balog et al. 2012]. For example, for many years TREC's Enterprise Search Track has included a track for researchers to empirically assess methods for expert finding [Craswell et al. 2005]. Methods proposed can generally be grouped into two categories according to the source of expertise indicators used. First, content-based methods utilize textual content and related documents that contain terms semantically relevant to the candidates' expertise areas. Several works adopt content-based approaches to identify the most appropriate community members for answering a given question in question-answering systems; for example, Bouguessa et al. [2008], Guo et al. [2008], and Pal et al. [2012]. Al-Kouz et al. leverage user profiles and posts to match topic expertise on Facebook [Alkouz et al. 2011]. Balog et al. proposed a candidate generative model that represents a candidate directly by terms and a document model that first finds documents that are relevant to the topic and then locates the experts associated with these documents [Balog et al. 2006]. Second, graph-based methods rely on social link analysis to consider each expert candidate's importance or social influence (e.g., Dom et al. [2003], Yeniterzi and Callan [2014], and Zhang et al. [2007a]). For example, Campbell et al. utilize the link between authors and receivers of emails to improve expert finding in an email-based social network [Campbell et al. 2003]. Moreover, there exist hybrid models considering both textual content and social relationships in expert finding (e.g., Bozzon et al. [2013], Wang et al. [2013b], and Weng et al. [2010]).

Recently, effort has focused on expert finding in Twitter-like systems. Weng et al. consider both tweet content and link structures among users to find topic experts on Twitter [Weng et al. 2010]. Based on the list meta-data in Twitter, Ghosh et al. built the Cognos systems to help find experts on a specific topic [Ghosh et al. 2012]. They rank experts by taking into account the overall popularity of a candidate and topic similarity. In the past year, a few efforts have begun to examine local aspects of expertise finding [Cheng et al. 2014; Li et al. 2014]. Li et al. investigate expertise in terms of a user's knowledge about a place or a class of places [Li et al. 2014]. Cheng et al. identified several factors, including local authority and topical authority, for assessing local expertise [Cheng et al. 2014]. Note that the LocalRank method in Cheng et al. [2014]—which considers topical authority and local authority—is similar in spirit to Cong et al. [2009], in which a rank scoring function is defined over a combination of document relevance and physical distance. Experimentally, we observe that LExL outperforms LocalRank, indicating the importance of these additional models that build on what was proposed

[Cheng et al. 2014; Cong et al. 2009]. Compared to these works, we introduce the first learning-based method for ranking local experts, introduce a new distance-biased random walk class of feature that models distance and relevance jointly (rather than independently), and conduct the first comprehensive study of factors impacting local expert ranking. Alternatively, many commercial systems provide search capabilities over regional content (e.g., Google+ Local, YellowPages.com, Craigslist) but not direct access to local experts, nor transparent models of how (or even whether) user expertise is assessed.

The method in this paper builds on learning-to-rank [Liu 2009], which has been an active area of ranking that builds on results in machine learning. Generally, learning-to-rank can be classified into three main types: pointwise methods, in which a single score is predicted for each query document through solving a regression problem; pairwise methods, in which the relative quality of each document pair is judged in a binary classification problem; and listwise methods, in which the evaluation metric is optimized as the direct goal [Hang 2011]. In the area of expert finding, Yang et al. [2009] apply Ranking SVM to rank candidate experts for ArnetMiner, an academic web resource. Moreira et al. explore the use of learning-to-rank algorithms for expert search on DBLP [Moreira et al. 2011].

3. LEARNING APPROACH TO LOCAL EXPERT FINDING

In this section, we introduce the learning approach framework for finding local experts—**LExL**: **L**ocal **Ex**pert **L**earning. Given a query composed of a topic and a location, the goal of LExL is to identify high-quality local experts.

3.1. Problem Statement

We assume there is a pool of local expert candidates $V = \{v_1, v_2, \ldots, v_n\}$, each candidate is described by a matrix of topic-location expertise scores (e.g., column i is College Station, while row j is "web development"), and that each matrix element indicates to what extent the candidate is an expert on the corresponding topic in the corresponding location. Given a query q that includes both a topic t and a location l, our goal is to find the set of k candidates with the highest local expertise in query topic t and location l. For example, find the top experts on $t_q =$ "web development" in $l_q =$ College Station, TX. Note that the query location l can be represented at multiple granularities (e.g., a city name, a latitude-longitude coordinate). This location indicates the region of interest for the query issuer and so is not constrained to the home location or current position of this query issuer; for example, a user in San Francisco may issue a local expert query for Houston before visiting on a business trip.

3.2. Overview of Approach

To tackle the local expert finding problem, we propose a geo-spatial approach that integrates geo-crowd knowledge about each candidate with a learning-to-rank framework. Concretely, we exploit the crowd wisdom embedded in millions of geo-located Twitter lists, coupled with a learning framework to isolate the critical features that are correlated with local expertise.

Geo-Located Twitter Lists. A Twitter list allows an individual on Twitter to organize who she follows into logical lists. For example, Figure 1 shows one list named "Food!" that contains 14 Twitter accounts including Alton Brown, Mind of a Chef, and America's Test Kitchen. Collectively, Twitter lists are a form of crowd-sourced knowledge whereby aggregating the individual lists constructed by distinct users can reveal the crowd perspective on how a Twitter user is perceived [Ghosh et al. 2012]. In this article, we exploit the geo-social information of 13 million lists—provided to us by the authors of

Fig. 1. Twitter List Example.

Table I. Geo-tagged Twitter List Data

Data Type	Total # of Records
Lists	12,882,292
User List Occurrences	85,988,377
Geo-Tagged List Relationships	14,763,767

Cheng et al. [2014]—in which we have the fine-grained location information of both the list creator (or *labeler*) and the member of the list (or *labelee*). In total, there are 86 million user occurrences on these lists, of which we have 15 million geo-tagged list relationships. So we know, for example, that Alice from Houston has labeled Bob from College Station as a Foodie. Thus, the aggregate list information may reveal not just the general crowd perspective on each user, but also the *local crowd's perspective*. High-level statistics of the dataset are listed in Table I. Further details of the dataset collection method can be found in Cheng et al. [2014]. In addition to this list information, we also crawl the content associated with these users for the period of May 2015 to September 2015.

3.3. Learning Approach

A previous approach by Cheng et al. [2014] focused on the local expert ranking problem using a linear combination of topical authority and local authority. In that work, topical authority was designed to capture the candidate's expertise on a topic area (e.g., how much does this candidate know about web development?). They adopted a language modeling approach [Balog et al. 2006] adapted to Twitter lists, where each candidate was described by a language model based on the Twitter list labels that the crowd has applied to them.

Local authority was designed to capture a candidate's authority with respect to a location (e.g., how well does the local community recognize this candidate's expertise?). Several approaches were suggested, including one that measured the average distance spread of list labelers to a candidate with respect to a query location—so that candidates who were listed by many people in an area of interest (e.g,. Joe has been labeled by 100 people from College Station) would be considered locally authoritative (e.g., Joe is well-recognized in College Station). These two aspects of local expertise—topical authority and local authority—were combined in a linear fashion to arrive at an overall score for each candidate.

More generally and in the presence of ground truth training data (see the evaluation setting in Section 5.2), we propose to transform the local expert ranking problem from an unsupervised linear combination of local authority and topical authority into a supervised learning-to-rank framework that can combine any number of local expertise

features, using a tool such as LambdaMART [Burges et al. 2011; Wu et al. 2008]. While we experiment with four different learning to rank algorithms in the experiments, we focus our discussion here on LambdaMART as a representative learning-to-rank framework (which we find experimentally has the best performance). LambdaMART is an instance of Multiple Additive Regression Tree (MART), which is based on the idea of boosting. It trains an ensemble of weak regression tree models and then linearly combines the prediction of each one of them into a final model that is stronger and more accurate. In ranking tasks, the measures that are typically optimized include NDCG, MAP, or MRR. Unfortunately, gradient boosting is not suitable if we directly consider these metrics as loss functions since they are not differentiable at all points. Hence, LambdaMART tunes the parameters of the regression trees using a variable λ, indicating whether the documents should be moved up or down in the rank list as the gradient of parameters based on the evaluation metric used, such that the evaluation metric is optimized directly while learning. In our experiments, we adopt NDCG as the evaluation metric.

4. FEATURES FOR LOCAL EXPERTISE

In this section, we describe five classes of features that potentially contribute to local topic expertise of a user: user-based features, tweet content features, list-based features, local authority features, and distance-biased random walk features. The user-based, list-based, and local authority features capture key characteristics of users and locations of interest in assessing the local topic expertise of our candidates. Compared to previous work that has only used direct keyword match to find relevant users (e.g., a query for "sports" matches a user who has been labeled with "sports"), we propose to integrate richer topical information from content that users have actually posted (the *tweet content* features). The last class of features—*distance-biased random walk*—is especially important because these features naturally integrate expertise propagation into a framework that models the query location, candidate location, and the location of list labelers. Compared to previous works, these distance-biased random walk features model candidates through two perspectives—both as labelers and as labelees—and directly embed distance into the transition probabilities to more robustly model the location preferences of users and candidates. In this work, we focus on 39 features, summarized in Table II.

4.1. User-Based Features

The first group of features captures user-oriented aspects that are independent of the query topic and query location. These features are simple measures of the popularity, activity, and longevity of each candidate and can be seen as crude first steps toward capturing how knowledgeable each user is:

—*User Network ($N_{follower}$, N_{friend})*: The first two features measure the number of followers that a candidate has, as well as the number of friends that this candidate has, where friends represent users who are both following and followed by the candidate.
—*User Activity (N_{fav}, N_{status})*: These two features are crude measures of a user's activity-level on Twitter, where N_{fav} captures the number of favorite tweets the user marked and N_{status} is the number of tweets posted by the user.
—*Longevity (T_{create})*: The final user feature is simply the UTC datetime when an account was created. In this way, the longevity (or freshness) of the user can be integrated into the ranking model.

Table II. List of Features Used for Ranking Local Expert Candidates

		User-based Features
1	$N_{follower}$	The number of followers this candidate has.
2	N_{friend}	The number of users this candidate is following.
3	N_{fav}	The number of tweets this candidate has favorited in the account's lifetime.
4	N_{status}	The number of tweets (including retweets) posted by the candidate.
5	T_{create}	The UTC datetime when the user account was created on Twitter.
		Tweet Content Features
6	twP_c	Avg. number of tweets that a candidate c posted in one week.
7	tw_H	Avg. tweet Entropy of a candidate c.
8-15	twB_t	Topic bayesian scores of a candidate c in 8 topics, t.
16-19	twB_l	Location bayesian scores of a candidate c in 4 topics, l.
		List-based Features
20	N_{listed}	The number of lists that this candidate appears on.
21	T_{listed}	The number of on-topic lists that this candidate appears on.
22	N_{list}	The number of lists this candidate has created.
23	T_{list}	The number of on-topic lists this candidate has created.
24	$list\_score_c$	The average quality of the lists that the candidate is a member of.
		Local Authority Features
25	d_c	Avg. distance from candidate c to all the users who appear on c's lists.
26	d_{ct}	Avg. distance from candidate c to all the users who appear on c's on-topic lists.
27	d_u	Avg. distance from candidate c to all the labelers of c.
28	d_{ut}	Avg. distance from candidate c to all the labelers whose on-topic list has c.
39	d_{uq}	Avg. distance from query location to labelers whose on-topic list has c.
30	d_{cq}	Distance between candidate c and the query location l_q.
31	$Prox_c$	Candidate Proximity, as defined in Section 4.4.
32	$Prox_{spread}$	Spread-based Proximity, as defined in Section 4.4.
		Distance-Biased Random Walk Features
33	ha_0	$\mathcal{A}^n(c) = p \cdot \sum_{u_i} \mathcal{H}^{n-1}(u_i) \cdot \frac{1}{\mathcal{O}(u_i)} + \frac{1-p}{N}$
		$\mathcal{H}^n(u) = p \cdot \sum_{c_i} \mathcal{A}^{n-1}(c_i) \cdot \frac{1}{\mathcal{I}(c_i)} + \frac{1-p}{N}$
34	ha_1	$\mathcal{A}^n(c) = p \cdot \sum_{u_i} \mathcal{H}^{n-1}(u_i) \cdot P_1(u_i, c) + \frac{1-p}{N}$
		$\mathcal{H}^n(u) = p \cdot \sum_{c_i} \mathcal{A}(c_i) \cdot \frac{1}{\mathcal{I}(c_i)} + \frac{1-p}{N}$
35	ha_2	$\mathcal{A}^n(c) = p \cdot \sum_{u_i} \mathcal{H}^{n-1}(u_i) \cdot \frac{1}{\mathcal{O}(u_i)} + \frac{1-p}{N}$
		$\mathcal{H}^n(u) = p \cdot \sum_{c_i} \mathcal{A}^{n-1}(c_i) \cdot P_1'(u, c_i) + \frac{1-p}{N}$
36	ha_3	$\mathcal{A}^n(c) = p \cdot \sum_{u_i} \mathcal{H}^{n-1}(u_i) \cdot P_2(u_i, c, l_q) + \frac{1-p}{N}$
		$\mathcal{H}^n(u) = p \cdot \sum_{c_i} \mathcal{A}^{n-1}(c_i) \cdot \frac{1}{\mathcal{I}(c_i)} + \frac{1-p}{N}$
37	ha_4	$\mathcal{A}^n(c) = p \cdot \sum_{u_i} \mathcal{H}^{n-1}(u_i) \cdot \frac{1}{\mathcal{O}(u_i)} + \frac{1-p}{N}$
		$\mathcal{H}^n(u) = p \cdot \sum_{c_i} \mathcal{A}^{n-1}(c_i) \cdot P_3(u, c_i, l_q) + \frac{1-p}{N}$
38	ha_5	$\mathcal{A}^n(c) = p \cdot \sum_{u_i} \mathcal{H}^{n-1}(u_i) \cdot P_2(u_i, c, l_q) + \frac{1-p}{N}$
		$\mathcal{H}^n(u) = p \cdot \sum_{c_i} \mathcal{A}^{n-1}(c_i) \cdot P_3(u, c_i, l_q) \frac{1-p}{N}$
39	ha_6	$\mathcal{A}^n(c) = p \cdot \sum_{u_i} \mathcal{H}^{n-1}(u_i) \cdot P_4(u_i, c, l_q) + \frac{1-p}{N}$
		$\mathcal{H}^n(u) = p \cdot \sum_{c_i} \mathcal{A}^{n-1}(c_i) \cdot P_4'(u, c_i, l_q) + \frac{1-p}{N}$

4.2. Tweet Content Features

Naturally, a more significant contributor to a candidate's perceived expertise is the content that user actually contributes to the community. Hence, we next consider a group of features that seek to summarize a candidate's content. Note that since tweets are inherently limited in size, we aggregate a candidate's posts over each week into a larger pseudo-document.

—*Posting Frequency (twP_c)*: The average number of tweets that the candidate c posted in one week.
—*Tweet Entropy (tw_H)*: The average entropy of a candidate's tweets in each week. This feature shows how informative the candidate's tweets are:

$$tw_H = -\sum_{j=1}^{n_{document}} \sum_{i=1}^{n_{term}} p(t_{ij}) \cdot logp(t_{ij}),$$

where $p(t_{ij})$ is the probability of a term t_i shows up in $document_j$.

—*Topic Bayesian Scores (twB_t)*: The posterior probability $P(topic_j|c)$, which represents the probability of $topic_j$ given candidate c. Here, we apply a naive Bayes method to get the posterior probability. We assume equal prior for topics. The observation $P(t_i|topic_j)$, probability that term t_i appears in $topic_j$, is learned from a corpus of topic-relevant pages crawled from Wikipedia.

—*Location Bayesian Scores (twB_l)*: Similar to Topic Bayesian Scores. The observation $P(t_i|l_j)$, probability that term t_i appears in location l_j, is trained from Wikipedia pages relevant to the location l_j.

4.3. List-Based Features

The third group of features extract expertise evidence directly from the Twitter list evidence but ignores the geo-spatial features of the lists (those aspects are part of the following two groups of features). Twitter lists have been recognized as a strong feature of expertise in previous work [Ghosh et al. 2012]. In particular, lists can shed light on a candidate from two perspectives:

—*Appearing on Lists (N_{listed}, T_{listed})*: On one hand, lists that a candidate appears on will reflect how that candidate is perceived by others. The aggregated information from all lists indicates how well the candidate is recognized.

—*Maintaining Lists (N_{list}, T_{list})*: On the other hand, lists the candidate creates (if any), reflect the candidate's personal interest, which may reflect his expertise. For example, a candidate with a list about food may himself be a foodie.

For these features, we consider all lists as well as a more focused group of on-topic lists (e.g., if the query is for "entrepreneurs," we only consider entrepreneur-related lists; these lists are selected by keywords matching). Moreover, we define a new feature to characterize the quality of a candidate's on-topic lists. This new feature—$list\_score_c$—is defined as:

$$list\_score_c = \frac{\sum_{i=1}^{N_{on\_topic}(c)} Q_{list}(i)}{N_{on\_topic}(c)}, \quad where\ Q_{list} = \frac{1}{k}\sum_{j=1}^{k} N_{on\_topic}(j),$$

where $Q_{list}(i)$ is the quality of i's list and $N_{on\_topic}(c)$ is the number of on-topic lists the candidate is in. Here, Q_{list} represents the average number of times each user in the list has been labeled with the topic of interest and k is the number of users in the list.

4.4. Local Authority Features

The fourth set of features focus on the local authority of a candidate as revealed through the geo-located Twitter lists. The main idea is to capture the "localness" of these lists. Intuitively, a candidate who is well-recognized near a query location is considered more locally authoritative. We measure the local authority of a candidate in multiple ways:

—*Candidate-List Distance (d_c, d_{ct})*: The first two features measure the average distance from candidate c to all the users who appear on c's lists. The main idea here is that a candidate is considered a local expert if she is closer to the people on the lists she maintains. We consider one version that captures all of the lists (d_c) and one that only considers on-topic lists (d_{ct}).

—*Candidate-Labeler Distance (d_u, d_{ut})*: The next two features measure the average distance from a candidate c to all the labelers of c, capturing the localness of the

people who have listed the candidate. Again, we consider one version with all lists (d_u) and one with on-topic lists (d_{ut}).

—*Candidate-Query Distance (d_{uq}, d_{cq})*: These two features measure distance from the query location. The first (d_{uq}) is the average distance from a candidate's labelers to the query location; labelers who are closer to the query location are considered more authoritative. The second (d_{cq}) is the distance from a candidate to the query location; candidates closer to the query location (regardless of whether they have been labeled by locals) are considered more authoritative.

In all cases, we measure distance using the Haversine distance, which gives the great-circle distance around the earth's surface. Apart from these six basic distance features, we also adopt two features used in a previous study of local experts [Cheng et al. 2014]: Candidate Proximity $Prox_c$ and Spread-Based Proximity $Prox_{spread}$:

$$Prox_c(c, l_q) = \left(\frac{d_{min}}{d(c, l_q) + d_{min}} \right)^{\alpha}$$

where $d(c, l_q)$ denotes the Haversine distance between the candidate c's location and the query location l_q, and we set $d_{min} = 100$ miles. In this case, $\alpha = 1.01$ indicates how fast the local authority of candidate c for query location l_q diminishes as the candidate moves farther away from the query location.

The Spread-Based Proximity captures the average "spread" of a candidate's labelers with respect to a query location:

$$Prox_{spread}(U_c, l_q) = \sum_{u \in U_c} Prox_c(u, l_q) / \mid U_c \mid,$$

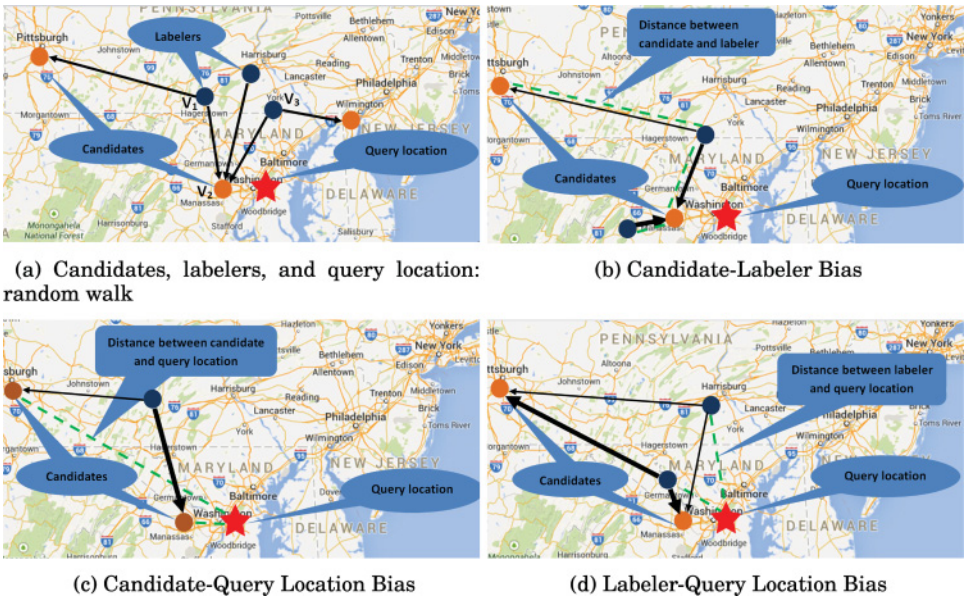
where u denotes one of the labelers U_c of candidate c. The "spread" measure considers how far an "audience" u is from the query location l_q on average. If the "core audience" is close to a query location on average, the candidate gets a high score of $Prox_{spread}$.

4.5. Distance-Biased Random Walk Features

Whereas the previous local authority features consider relationships between a labeler and a candidate, they only consider direct evidence. That is, only the one-hop distance is ever considered. We introduce in this section a set of features that incorporate additional network context beyond these one-hop relationships. Concretely, we explore features based on a random walk model that directly incorporates the location of interest (the query location), the location of a candidate expert, and the location of external evidence of a candidate's expertise (e.g., in the case of the Twitter lists, the location of the list labeler). The main intuition is to bias a random walker according to the distances between these different components (the query location, the labeler, the candidate) for propagating local expertise scores. In this way, each candidate can be enriched by the network formed around them via Twitter lists.

4.5.1. Baseline Random Walk over Twitter Lists. We begin by modeling a graph based on Twitter lists, which contains the set of users and labeler-candidate relations on topic t, as a directed graph $G = (V, E_t)$. The nodes $V = \{v_1, v_2, \ldots, v_n\}$ correspond to users. A directed edge $e = (v_1, v_2)_t$, where $e \in E_t$, indicates the presence of a labeler-candidate relation (which we will use listing for short) from user v_1 to user v_2 on topic $t \in T$. Here, $E_t = \{e_1, e_2, \ldots, e_m\}$ represents the set of listings on topic t and $T = \{t_1, t_2, \ldots, t_p\}$ is the set of topics. Furthermore, each user v_i has an associated location $l(v_i)$.

Over this graph, we can define a transition probability for a random walker to move from one user to the next, either by following forward links or by following backward links. As a baseline, consider a simple random walker model akin to Kleinberg's Hubs



(a) Candidates, labelers, and query location: random walk

(b) Candidate-Labeler Bias

(c) Candidate-Query Location Bias

(d) Labeler-Query Location Bias

Fig. 2. Three distance biases defined over random walk in Twitter list network.

and Authorities [Kleinberg 1999]. In Figure 2(a), suppose the walker starts from a user v_1, selects a particular topic t, and then randomly selects a member v_2 in the list to follow the outgoing link (or, as we say, forward link). The random walker further checks the topic t lists of which v_2 is member and reversely follows an incoming link (or, as we say, backward link) to user v_3 who has labeled v_2 in a topic t list. The random walker alternatively follows a forward link and a backward link in a strict manner and continues this process forever. We further incorporate additional randomness: At each step, the walker can either follow the links or jump to a random user $v_m \in U$. In summary, we can assign each candidate a local expertise authority score $A^n(c)$ (reflecting local expertise) and each labeler a local expertise hub score $H^n(u)$ (reflecting how well this labeler is a conduit to other local experts):

$$\mathcal{A}^n(c) = p \cdot \sum_{u_i} \mathcal{H}^{n-1}(u_i) \cdot \frac{1}{\mathcal{O}(u_i)} + \frac{1-p}{N}$$

and

$$\mathcal{H}^n(u) = p \cdot \sum_{c_i} \mathcal{A}^{n-1}(c_i) \cdot \frac{1}{\mathcal{I}(c_i)} + \frac{1-p}{N}$$

where p is the probability of following a link and $1-p$ is the probability of teleporting. $\mathcal{O}(u)$ is the outdegree of a labeler and $\mathcal{I}(c)$ is the indegree of a candidate.

4.5.2. Integrating Local Bias. The preceding random walk is defined for capturing overall authoritative (and hub-like) candidate, without regard for localness. Hence, we propose three approaches for directly integrating distance bias:

Candidate-Labeler Bias: The first approach is to increase the likelihood of following a link to closer candidates (or, conversely, to decrease the likelihood of following links to distant candidates). The intuition here is that a labeler may tend to have stronger knowledge of candidates who are closer, reflecting their local connection. For example,

in Figure 2(b), the probability of transitioning from a labeler to a candidate is higher for closer nodes (represented here by a thicker arrow). The transition probability is lower for more distant nodes (represented by a thinner arrow). Formally, we define these probabilities in the following way: Let $d(u, c)$ be the Haversine distance between labeler and candidate, and D_1 is a nonlinear mapping in the form of Candidate Proximity discussed in Section 4.4, which maps $d(u, c)$ to a real value in [0,1]. Other mappings are possible, but we find good results using this approach. The probability of following a forward link from u to c is defined as:

$$P_1(u, c) = \frac{D_1(d(u, c))}{\sum_{c_j:u\to c_j} D_1(d(u, c_j))}.$$

The probability of following a backward link from c to u is defined as:

$$P_1'(u, c) = \frac{D_1(d(u, c))}{\sum_{u_j:c\to u_j} D_1(d(u_j, c))}.$$

Candidate-Query Location Bias: The second approach is to increase the likelihood of following a forward link to a candidate who is closer to a query location (and, conversely, to decrease the likelihood of following links to candidates who are distant to the query location). The intuition is that a candidate who is closer to a query location is more likely to be knowledgeable about the topic at the query location. For example, in Figure 2(c), we see that the probability of transitioning to the candidate close to the query location is larger (represented by the thicker arrow) versus the probability of transitioning to the other candidate (represented by the thinner arrow). Formally, let $d(c, l_q)$ be the Haversine distance between the location of candidate and query location, and D_2 maps $d(c, l_q)$ to a real value to a real value in [0,1], where D_2 is a mapping similar to D_1. The probability of following a forward link from u to c is defined as:

$$P_2(u, c, l_q) = \frac{D_2(d(c, l_q))}{\sum_{c_j:u\to c_j} D_2(d(c_j, l_q))}.$$

Labeler-Query Location Bias: The third approach is to increase the likelihood of following a backward link to a labeler who is closer to the query location (and, conversely, to decrease the likelihood of following a backward link to a labeler who is distant to the query location). The intuition is that a labeler who is closer to a query location is more likely to label high-quality local expert candidates at the query location. For example, we can see in Figure 2(d) that the labeler who is closer to the query location has a higher probability associated with that edge (as represented by the thick arrow). Otherwise, the probability is small (as represented by the thin arrow). Formally, let $d(u, l_q)$ be the Haversine distance between labeler and query location, and D_3 maps $d(u, l_q)$ to a real value in [0,1], where D_3 is a mapping similar to D_1. The probability of following a backward link from c to u is defined as:

$$P_3(u, c, l_q) = \frac{D_3(d(u, l_q))}{\sum_{u_j:c\to u_j} D_3(d(u_j, l_q))}.$$

Combining Bias Factors: Finally, we can combine the different bias factors in various ways. For illustration, assume we want to combine all three bias factors: $\{d(c, l_q), d(u, l_q), d(u, c)\}$. We can define the probability to follow a forward link as:

$$P_4(u, c, l_q) = \frac{D_1(d(u, c)) \cdot D_2(d(c, l_q))}{\sum_{c_j:u\to c_j} D_1(d(u, c_j)) \cdot D_2(d(c_j, l_q))},$$

and, in the same fashion, we can get the probability P_4', which characterizes the probability of following a backward link.

Finally, we can embed these different distance-bias factors into the local expertise authority score $A^n(c)$ described earlier to generate a series of new features. Specifically, we generate seven new features based on the distance-biased random walk (as shown in Table II). ha_0 uses the original setting with no distance influence. ha_1 and ha_2 take the distance between labeler and candidate into account ($P_1(u, c)$ and $P_1'(u, c)$). ha_3 considers the distance between the location of a candidate and the query location ($P_2(u, c, l_q)$). ha_4 considers the distance between the labeler and the query location ($P_3(u, c, l_q)$). ha_5 considers how distant a candidate ($P_2(u, c, l_q)$) and the labelers ($P_3(u, c, l_q)$) are from the query location. Finally, h_6 considers the distance between each pair of the three entities ($P_4(u, c, l_q)$).

5. EVALUATION

In this section, we present the experimental setup, including the collection of ground truth data via AMT, alternative local expert ranking methods, and metrics for comparing these methods. We then report on a series of experiments designed to answer the following questions: How does the learning-based local expert ranking approach compare to existing methods? How stable are the results across different topics and locations? What features are most important for identifying local experts? Can a local expert model trained on one topic generalize to other topics?

5.1. Experimental Setup

Our experiments rely on the dataset described in Section 3.2, totaling 15 million geotagged list relationships.

Queries. We adopt a collection of eight topics and four locations that reflect real information needs. The topics are divided into broader local expertise topics—"food," "sports," "business," and "health"—and into more specialized local expertise topics that correspond to each of the broader topics—"chefs," "football," "entrepreneurs," and "healthcare." The locations are New York City, San Francisco, Houston, and Chicago, which all have relatively dense coverage in the dataset for testing purposes.

Retrieving Candidates. For each method tested, we retrieve a set of candidates for ranking based on topics derived from list names. For each list name, we apply tokenization, case folding, stopword removal, and noun singularization. We separate string patterns like "FoodDrink" into two tokens "food" and "drink." We consider each of the remaining keywords as a *topic*. Finally, each candidate is associated with all topics derived from this process, resulting in a set of potential candidates to be ranked.

Proposed Method: Local Expert Learning (LExL). There are a wide variety of learning-to-rank approaches possible; in this article, we evaluate four popular learning-to-rank strategies: Ranknet, MART, Random Forest, and LambdaMART. We use an open source implementation of these methods in the RankLib toolkit. While we previously introduced LambdaMART, here we briefly introduce these three other variations of LExL. Ranknet is a pairwise learning-to-rank method in which each pair of the candidates is considered together to form a positive or negative instance. The cost function of Ranknet aims to minimize the number of inversions in ranking. MART is based on the idea of boosting, and it uses gradient-boosted decision trees for prediction tasks. The prediction model is a linear combination of the outputs of a set of regression trees. Random Forest is an application of bagging, which can substantially improve the quality of probability estimates in almost all domains [Provost and Domingos 2003]. However, bagging has two disadvantages: greater computation cost and loss of

comprehensibility. Note that LambdaMART is a specific instance of MART that evolved out of a combination of Ranknet and MART. For each topic, we randomly partition the collected candidates together with their five categories of features into four equal-sized groups for training and testing. We use four-fold cross-validation for reporting the results. We compare our proposed approach with two state-of-the-art approaches for finding local experts:

—**Cognos+ [Ghosh et al. 2012]:** The first baseline method is the Cognos expert ranking scheme. Cognos was originally designed for identifying general topic experts, so the ranked lists from Cognos are independent of query location. Hence, we modify Cognos by incorporating a distance factor when calculating cover density ranking [Clarke et al. 2000], where each label is weighted by a distance factor range [0,1], similar to Candidate Proximity discussed in Section 4.4. We refer to this location-sensitive version of Cognos as Cognos+.

—**LocalRank [Cheng et al. 2014]:** The second baseline method is the LocalRank framework proposed in Cheng et al. [2014]. This framework ranks candidates by a linear combination of local authority and topical authority. We choose the best performing combination reported in that paper—spatial proximity plus direct labeled expertise (SP+DLE)—as the baseline to compare against.

Note that of these alternative methods are unsupervised, whereas the learning-based approach proposed here integrates labeled training data to bootstrap the ranker. Naturally, we would expect the supervised approach to perform well; our goal here is to measure this improvement as well as investigate the key factors for this improvement.

5.2. Gathering Ground Truth

Since there are no publicly available data that directly specify a user's local expertise given a query (location + topic), we rely on an evaluation based on ground truth by employing human raters (turkers) on Amazon Mechanical Turk to rate the level of local expertise for candidates via Human Intelligent Tasks (HITs).

5.3. Pooling Strategy

It is too expensive to manually label the local expertise of every candidate with each query pair (location + topic). Moreover, many candidates are irrelevant to the query location and do not possess expertise on the topic of interest. Hence, a pooling strategy is adopted to improve the effectiveness of obtaining relevance judgments by reducing the number of irrelevant candidates presented to turkers to improve their effective utilization [Kazai et al. 2011]. To build the pool of local expert candidates, the candidate set is sampled for each query pair, which only considers those candidates who appear at least once on on-topic list. 100 candidates are selected for each query pair, and then they are randomly assigned to different HITs.

5.4. HIT Design

Each HIT includes instructions and examples of local expertise along with 12 candidates to judge. Turkers can access the information about the query pair and a link to a candidate's Twitter page including account profile, recent tweets, lists, and home location. Turkers are then asked to rate each candidate's local expertise on a five-point scale corresponding to no local expertise (0), difficult to tell (1), a little local expertise (2), some local expertise (3), and extensive local expertise (4). The topic and location are kept the same within a single HIT, so the turkers can become familiar with the style of HITs with the task and make more consistent judgments. Several evaluation criteria [Kazai et al. 2011] are adopted to collect high-accuracy and reliable ground truth judgments. First, 2 out of the 12 candidates are set as trap questions, where we have

Table III. Turker Agreement for Topics

Topic	Accuracy	κ value
food	0.6845	0.5320
sport	0.7889	0.4903
business	0.7119	0.4596
health	0.7639	0.4461
chef	0.6640	0.4679
football	0.7758	0.3834
entrepreneur	0.7128	0.2868
healthcare	0.7675	0.5952
Average	0.7337	0.4576

already judged the candidates as either clearly local experts (4) or obviously having no local expertise (0). These trap candidates are chosen to identify turkers who give random judgments or make judgments only by the candidate's home location (e.g., quickly assigning high scores to those candidates whose locations are given as Houston on the map instead of looking at their Twitter information for a task seeking local experts on Houston healthcare). We also maintain a turker qualification type in AMT that only allows turkers whose results are consistently of good quality to continue working on our HITs. For each candidate, we collect five judgments from distinct turkers, and the majority judgment is taken as the final local expertise rating; if there is a tie in the vote, the ceiling of the average is taken as the final rating.

5.5. Turker Agreement

After running the HITs experiments, 16k judgments were collected in total across the eight topics and four locations based on the preceding settings. But are these assessments of local expertise reliable? To answer this, the *accuracy* and the *kappa statistic* [Fleiss et al. 1969] are calculated to explore the validity of turker judgments. The accuracy for a candidate c given a query pair q is defined as

$$Accuracy(c, q) = \frac{No.\ of\ majority\ judgments}{No.\ of\ judgments.}$$

Accuracy ranges from 0 to 1, with 0 meaning every judgment for the candidate is unique and agrees with no other judgment, and 1 meaning all raters give a consistent judgment for the candidate. The kappa statistic also measures interrater reliability, ranging from 0 to 1, with larger values indicating more consistency in judgments. In Table III, we show the accuracy and kappa values for each topic, where we treat local expertise scores of 2, 3, and 4 as relevant and scores of 0 and 1 as irrelevant. The average accuracy across all topics is 0.74, which indicates that around 3 out of 4 raters agree on whether one candidate is a local expert. The accuracy is higher in some topics (e.g., football), indicating that assessing local expertise may be inherently easier in some cases. For kappa, an average of 0.46 means "moderate agreement." As in the case of accuracy, there is variability in the scores, with the topic "entrepreneur" being the most controversial topic to judge and "healthcare" being the easiest.

5.6. Evaluation Metrics

To evaluate the quality of local expertise approaches, three metrics are adopted across all experiments: Rating@k, Precision@k, and NDCG@k.

Rating@k measures the average local expertise rating for a query pair to output the top-k experts for each approach, defined as:

$$Rating@k = \sum_{i=1}^{k} rating(c_i, q)/k,$$

where c is candidate and q is the query pair. In our scenario, k = 10. The Rating@10 here ranges from 0 to 4, where a value of 4 says the majority of the raters believe that every one of the top 10 experts found by the local expertise method has extensive local expertise. Since Recall will calculate the fraction of relevant ratings that are retrieved based on all Turkers' judgments across query topics and locations, the value of Recall won't change for different learning methods and different sets of features. Thus we utilize Rating@k instead of Recall.

Precision@k measures the percentage of the top-k suggested local experts who are actually local experts. Here, candidates with a rating 3 or 4 are considered relevant; all others are irrelevant. Note that this is a more conservative approach than the one for interjudge reliability; we want more distinguishing power deployed between approaches for comparing local expertise methods.

$$Precision@k = \sum_{i=1}^{k} r_i/k,$$

where $r_i = \begin{cases} 1 & \text{if rating}(c_i,\text{q}) \geq 3 \\ 0 & \text{else} \end{cases}$

NDCG@k compares how close each method's top-k ranking order of local experts is to the ideal top-k ranking order.

$$NDCG@10 = \frac{DCG@10}{IDCG@10},$$

where $DCG@10 = \sum_{i=1}^{10} \frac{2^{rating_i}-1}{\log_2(i+1)}$ and $IDCG@10 = \sum_{i=1}^{10} \frac{2^{rating_i'}-1}{\log_2(i+1)}$. $rating_i$ represents the actual rating of the candidate in position i, and $rating_i'$ represents the rating of the candidate in position i given the ideal decreasing ranking order of all candidates. $DCG@10$ is the Discounted Cumulative Gain (DCG) of the learned ranking order until position 10, and IDCG@10 is the maximum possible DCG up to position 10.

5.7. Results

5.7.1. Comparison versus Baselines. We begin by comparing the proposed learning method (LExL) versus the two baselines. Figure 3 shows the Precision@10, Recall@10, and NDCG@10 of each method averaged over all queries.[1] We consider the LambdaMART version of LExL, in addition to methods using Ranknet, MART, and Random Forest. First, we observe that three versions of LExL clearly outperform all alternatives, resulting in a Precision@10 of more than 0.76, an average Rating@10 of around 3.1, and an NDCG of around 0.84.

Cognos has been shown to be effective at identifying general topic experts. However, we see here that even a modified version that includes a distance factor is not compatible with local expert finding. For example, Cognos may identify a group of "healthcare" experts known nationwide, but it has difficulty uncovering local experts.

[1]Note that the results reported here for LocalRank differ from the results in Cheng et al. [2014] because the experimental setups are different. First, our rating has 5 scales, which are intended to capture more detailed expertise levels. Second, Cheng et al. [2014] only considers ideal ranking order for the top 10 results from LocalRank when calculating IDCG@10, whereas we consider a much larger corpus.

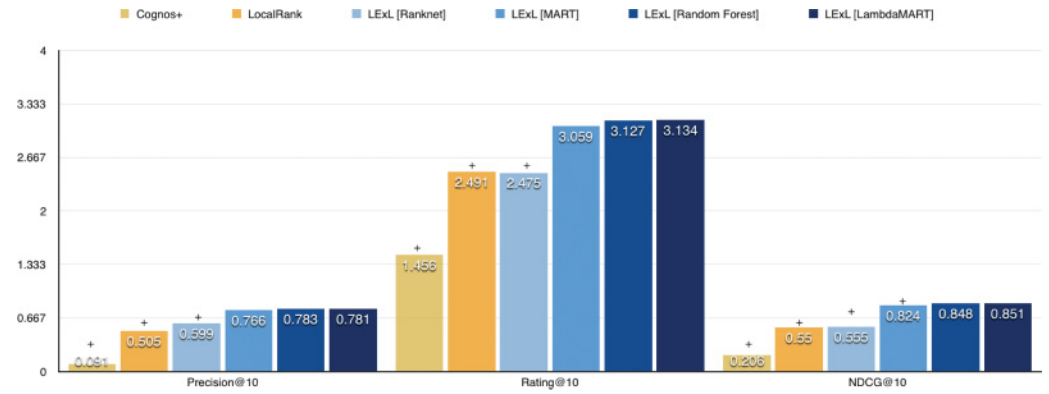


Fig. 3. Evaluating the proposed learning-based local expertise approach versus two alternatives. "+" marks statistically significant difference with LExL[LambdaMART] according to paired t-test at significance level 0.05.

LocalRank has a much better Precision@10 of around 0.5 compared to Cognos+, which indicates that 50% of the candidates it identifies have at least "some local expertise" for the query. The average Rating@10 is 2.49, which means the candidates are generally rated between "a little expertise" and "some expertise." Since LocalRank explicitly builds on both topical and local signals (by exploiting the distance between a candidate's labelers and the query location), it performs much better than Cognos+. However, LocalRank is only a linear combination of these two factors and so does not exploit either additional factors (like the random walk presented in this article) or take advantage of a learning approach for optimizing the weighting of these factors.

For the four LExL approaches, Ranknet performs comparably to LocalRank, but the remaining three all result in significantly better performance, with both Random Forest and LambaMART achieving comparably good results. These two methods have a Rating@10 of around 3.1, indicating that the local experts discovered have from "some local expertise" to "extensive local expertise." The Precision@10 and NDCG@10 also support the conclusion that these learning-based methods result in high-quality local experts. Since LambdaMART is significantly less computationally expensive (∼1/6 of the computing time of Random Forest), we adopt it for the remainder of the article.

5.7.2. Effectiveness Across Topics and Locations. Given the good performance of LExL with LambaMART, we next turn to comparing the effectiveness of this approach across the four general topics and four narrower topics before turning to a location comparison in the following discussion. Is the effectiveness of local expert finding consistent across topics? And does it vary by the specificity of the topic?

We observe in Table IV that NDCG@10 is consistently high for the four general topics, with an average value of 0.8212. Precision@10 and Rating@10 are also consistent for general topics except for the topic of "health," which has relatively low values. We attribute this poor showing due to data sparsity: (i) First, through manual inspection, we find that there are inherently only a limited number of candidates with high local expertise for the "health" topic in the training and testing datasets. (ii) Second, since we only consider candidates with "some local expertise" and "extensive local expertise" as good matches for a query, this additionally reduces the number of possible local experts. However, since the learning framework is effective at identifying even those few local experts in "health," we see a high NDCG@10.

We observe comparable results for the four narrower topics. The Precision@10 is lower than for the general topics (0.75 versus 0.81), but the NDCG@10 is higher (0.88

Table IV. Quality of Local Expert Rankings Across Topics

Topics	Precision@10	Rating@10	NDCG@10
food	0.8250	3.125	0.7442
sports	0.9152	3.225	0.9054
business	0.9237	3.368	0.8506
health	0.5873	3.059	0.7847
chefs	0.8233	3.163	0.9044
football	0.7283	2.933	0.9002
entrepreneurs	0.7377	3.040	0.7489
healthcare	0.7100	3.166	0.9673
General topic AVG	0.8128	3.193	0.8212
Subtopic AVG	0.7498	3.075	0.8802

Table V. Quality of Local Expert Ranking in Different Locations

Locations	P@10	R@10	NDCG@10
Houston	0.7214	2.917	0.8473
Chicago	0.7788	3.244	0.8486
New York	0.7875	3.294	0.8501
San Francisco	0.7563	3.081	0.8580

Table VI. Quality of Local Expert Ranking Using Different Sets of Features

Features	Precision@10	Rating@10	NDCG@10
User-based	0.6714†	2.748†	0.6955†
Tweet Content	0.6804†	2.730†	0.6973†
List-based	0.6839†	2.807†	0.6824†
Local Authority	0.7386†	3.002†	0.7662†
DistBRW	0.7431†	2.995†	0.7734†
All Features	0.7813	3.134	0.8507

'†' marks statistically significant difference with LExL[LambdaMART] according to paired T-test at significance level 0.05.

versus 0.82). Part of the higher NDCG results may be attributed to the decrease in the denominator of NDCG for these narrower topics (the Ideal DCG), so the ranking method need only identify some of a pool of moderate local experts rather than identify a few superstar local experts.

In a similar fashion, we evaluate the quality of LExL across the four query locations, as shown in Table V. For the most part, the Precision@10, Rating@10, and NDCG@10 show good consistency across these four locations—Chicago, Houston, New York and San Francisco—suggesting the potential of a learning-based method to identify factors associated with each location for uncovering local experts.

5.8. Evaluating Feature Importance

Given the strong performance of the learning approach for local experts, what is the significance of the different kinds of features used for learning? Recall that the learning model is built on five kinds of features—user-based, tweet content, list-based, local authority, and distance-biased random walks (DistBRW). To assess the importance of these different features, we train four different LExL models, one for each feature type. For example,the model is trained only using user-based features and then evaluates the quality of local experts identified.

We can see from Table VI that the five feature classes result in varying levels of local expert quality. The user-based, list-based, and tweet content features perform relatively well (especially when compared to LocalRank), although not as well as the

Table VII. Accumulated Times of Features Selected by Different Methods

Feature	REF	Tree-based	Feature	REF	Tree-based
$N_{follower}$	0	2	N_{listed}	1	2
N_{friend}	1	0	T_{listed}	4	2
N_{fav}	0	0	N_{list}	3	0
N_{status}	0	0	T_{list}	5	1
T_{create}	2	0	$list\_score_c$	1	6
			d_c	1	0
twP_c	0	0	d_{ct}	3	0
tw_H	0	0	d_u	3	1
$twB_{business}$	1	0	d_{ut}	4	7
$twB_{entrepreneur}$	0	0	d_{uq}	2	1
twB_{food}	1	1	d_{cq}	8	8
twB_{chef}	0	0	$Prox_c$	7	8
twB_{sport}	0	0	$Prox_{spread}$	6	6
twB_{food}	1	1	ha_0	2	5
twB_{health}	0	0	ha_1	5	4
$twB_{healthcare}$	0	0	ha_2	2	4
twB_{chi}	0	0	ha_3	2	2
twB_{hou}	0	0	ha_4	7	6
twB_{ny}	0	0	ha_5	2	6
twB_{sf}	0	0	ha_6	8	7

local authority and DistBRW features. These results suggest that intelligent combinations of many features via a learning method can outperform a simple combination of two carefully selected features (as in LocalRank). The DistBRW features achieve the highest Precision@10 and NDCG@10 among all five kinds of features. We attribute the results to DistBRW integrating expertise propagation as well as distance bias factors into capturing local expertise. We can observe that the combination of all features performs the best of all. One more interesting finding is that Tweet Content features didn't perform as well as we expected. This may be because the keywords related to topic and location in tweet content didn't appear very often. For example, an NFL quarterback may share his feelings about life more often than his expertise topics of football. Thus, we can conclude that tweet content is not the determining factor of finding local experts.

But which specific features are most informative, regardless of feature category? Here, we adopt two different feature selection methods to identify the most informative features for local expert ranking.

—**Recursive Feature Elimination (RFE).** In this approach, a linear regression model is trained and weight is assigned to each feature. Then, features with the smallest absolute weight are eliminated. For each topic, we keep eliminating the unimportant features until only a required number of features are left.
—**Tree-Based Feature Selection.** In this approach, a number of randomized decision trees are built on various subsamples of the dataset. The importance of a feature is determined by *Gini importance* or *Mean Decrease Impurity*, which is defined as the total decrease in node impurity of all trees in the ensemble [Breiman et al. 1984]. Features that attach to a node with higher *Gini importance* are more informative in the model.

Table VII shows the accumulated number of times that each feature is selected by the two different feature selection methods. For 39 features, most of the top features are from local authority features and DistBRW features.

The results is aggregated across all queries (topics + locations) and reported in Table VIII, which shows the top features for each feature importance method. There are seven common top features across both methods, which is highly consistent. Recall that d_{cq} and d_{ut} capture the distance from candidate to query location and the average

Table VIII. Individual Feature Importance

Method	Top-10 Features
RFE	d_{cq}, ha_6, $Prox_c$, ha_4, $Prox_{spread}$, T_{list}, ha_1, T_{listed}, d_{ut}, N_{list}
Tree-based	d_{cq}, $Prox_c$, ha_6, d_{ut}, ha_4, $Prox_{spread}$, ha_5, $list\_score_c$, ha_0, ha_1

Table IX. Performance Using Selected Features

Method	Precision@10	Rating@10	NDCG@10
RFE	0.7612	3.059	0.8357
Tree-based	0.7753	3.078	0.8444
All Features	0.7813	3.143	0.8507

distance from candidate to all on-topic labelers, respectively. $Prox_c$ shows the distance between the location of candidate and query location, and ha_6 is the steady-state local expertise authority score of a candidate incorporating all three distance influences. Based on the selection results, we can say that, compared to the tweet content, a candidate's list and location-related features are more decisive for finding local experts.

Ultimately, how explanatory are these features? We further train two additional LExL models—one using the top-10 features from the RFE feature importance method and one using the top-10 features using the tree-based method. Table IX shows the evaluation metrics for these two approaches versus the full-blown LExL model with all features. We observe that the difference of Precision@10 and Rating@10 is about 0.02 and NDCG@10 is about 0.1 for both methods, compared to the All Features case. Moreover, the difference between the values of three evaluation metrics and those of the All Features' case is not statistically significant. These results confirm the importance of the local authority features and the DistBRW features and further show that high-quality local expert models may be built using fairly compact features.

5.9. Generalizability of Local Expert Models

Finally, can a learned model be reused to rank other topics? The generalizability of the local expert models is explored in this section. In many cases, we can imagine building a local expert model that is optimized for one type of topic (e.g., healthcare) but then we want to apply the model to a different topic (e.g., finance), for example, in cases where training data is unavailable or expensive to collect. Are the models of local expertise generalizable enough to support high-quality local expert finding in these new topic areas? Or do the key features and feature weightings vary from topic to topic, so that a specialized model must be built for each topic?

The first experimental setup here is to train a model on each of four topics and then to apply this model to a different topic. Concretely, we train over the four general topics—health, food, business, and sports—and then rank candidates in each of the four narrower topics—healthcare, chefs, entrepreneurs, and football. Intuitively, a model trained over a related topic is expected to perform better than a model trained over a less similar topic (e.g., a health-based local expert model should do better for healthcare but worse for football). But does this hold? And how well do the less related models perform?

The results of this experiment are shown in Table X. For each of the four narrower topics, the model corresponding to the most related general topic indeed produces the best results. Perhaps surprisingly, these models perform on par with the models trained over the individual topics in Table IV or even better in Precision@10 and Rating@10. Since the general topic models build a broader measure to define a local expert than the individual models of a narrower topic, they can rank the more related candidates higher, which leads to high Precision@10 and Rating@10.

Table X. Applying a Model Learned on One Topic to Rank Local Experts on a Different Topic

Topic	Model	Precision@10	Rating@10	NDCG@10
chefs	health	0.8312	3.094	0.7363
	food	**0.8738**	**3.152**	**0.8012**
	business	0.8250	3.131	0.6644
	sports	0.8687	3.075	0.7963
	general	0.8633	3.125	0.8112
	original	0.8233	3.163	0.9044
football	health	0.6987	2.781	0.7910
	food	0.6125	2.618	0.5718
	business	0.6437	2.731	0.5934
	sports	**0.7083**	**2.875**	**0.8026**
	general	0.7014	2.925	0.8473
	original	0.7283	2.933	0.9002
entrepreneurs	health	0.7033	2.816	0.6247
	food	0.7166	2.825	0.6018
	business	**0.7511**	**2.950**	**0.7144**
	sports	0.7433	2.841	0.6432
	general	0.7866	2.966	0.6906
	original	0.7377	3.040	0.7489
healthcare	health	**0.7037**	**3.118**	**0.8844**
	food	0.6687	2.975	0.6443
	business	0.6875	3.002	0.7570
	sports	0.6650	3.090	0.8229
	general	0.6833	3.050	0.8847
	original	0.7100	3.166	0.9673

Even for models built on very different topics, we see encouraging results. For example, the sports-based model for ranking chefs results in Precision@10 of 0.86, Rating@10 of 3.1, and NDCG@10 of 0.80, and the health-based model for ranking football results in Precision@10 of 0.69, Rating@10 of 2.8, and NDCG@10 of 0.79. These results indicate the potential of learning models that can be extended to new local expert topics.

In the second experiment, instead of having a model for each general topic, we train a single model for four general topics altogether and then test this model on each subtopic. Table X shows that the general model performs no worse than each individual model. This is attributed to more training data and avoidance of overfitting to one topic. It indicates that we may find a common local expert model that is applicable regardless of the specific topic.

6. LEXL SYSTEM DESIGN

We conclude with a discussion of issues impacting the deployment of LExL. The discovered local experts can be integrated into a variety of systems—including (i) recommender systems that can be biased to prefer the opinions and rating of the discovered local experts; (ii) local question-answer systems, where local queries are routed to local experts; and (iii) locally flavored information streams like the Facebook newsfeed and the Twitter stream, where posts can be re-ranked based on the local expertise scores of who is posting, among many other possible application scenarios. In the following discussion, we raise some key points that can impact the success of deploying LExL into these scenarios, including efficiency, incentives, and some domain-specific concerns.

6.1. Efficiency

First, the complexity of learning a ranking model using the LambdaMART algorithm is $O(m \cdot N^2)$, where N is the number of training samples and m is the number of trees. As we have seen through experiments, a generalized ranking model can be built that performs competitively for other query locations and query topics, meaning that

one strategy would be to build models entirely offline. In this way, we could perform rigorous offline optimization (including identifying key sets of powerful features), such that the real-time assessment of new candidates would require only the generation of features for that candidate and then ranking the candidates according to the model-given features. For each query topic, we should keep track of query-relevant users, together with their list features. To update the system with new expert candidates, we can collect users together with their features at regular time intervals to refresh the candidates. Most of the features can be acquired continuously because they can be pre-computed offline (e.g., user-based features, the majority of the local authority features, and tweet content features). For the distance-biased random walk features, all seven can be updated simultaneously, and we empirically set the number of iterations to a constant c that is less than 10, for the algorithm to converge. Hence, the complexity for feature construction is thus $c \cdot O(n)$, where n represent the number of candidates. The ranking itself takes $O(n \cdot logn)$. Overall, the complexity is $O(n \cdot logn)$.

For application scenarios that require real-time local expert discovery, there are many ways to further decrease the response time required. For example, we could selectively lower the location accuracy and report results for nearby cities. Separately, we could cache results for some frequent query locations so that query issuers can get immediate responses. Additionally, we can set a distance threshold such that distant candidates are not considered to decrease the computation cost for feature collection and ranking.

6.2. Incentives

A second key concern is incentives. In some application scenarios—for example, re-ranking an information stream or recommending based on local experts—we can mine evidence of local experts without directly engaging with them. In other scenarios—for example, a local expert-powered question-answer system—incentivizing local experts to participate is critical. Indeed, we have seen a wide variety of efforts that target incentive schemes in crowd-sourcing systems. For example, some systems are built around tasks that are inherently fun—like the ESP game and peekaboom—to encourage participation. Some sites, like Quora, Stackoverflow, and Wikipedia, can attract and retain participants for several reasons: First, the inherent inclination toward human interaction and community work on similar things creates emotional bonds. Second, on the one hand, users have a sense of accomplishment because their knowledge and experience are of immediate use and value to the individual participants, and, on the other hand, by supporting the system for others, the system will likely return the favor in an area that users need. Third, the motivation for sites like Wikipedia mainly comes from one's academic interests and charity, and some people ask for no return, simply for the sake of its success. Of course, the most direct and effective approach is offering rewards. For example, some systems provide a badging system or similar accumulation of reputation points that can serve to incentivize participation. Money is a great motivator for prominent crowd-sourcing platforms like Amazon Mechanical Turk and Crowdflower.

In our continuing work, we are interested in combining several of these features to encourage local experts to participate in a prototype question-answer system. Concretely, we are interested in making answering a combination of reward-motivated and self-motivated benefits. When people have some instant information needs, they are willing to pay money and get a precise and informative response quickly. Also, the award money will encourage experts to answer. Meanwhile, we are also expecting to see a more natural eco-system where users are eager to help each other and seek and contribute knowledge.

6.3. Question-Answer Issues

A local expert question-answer system will require additional system design. For example, how do we properly match the question with the appropriate level of expertise? How do we match similar questions so that previously answered questions can be reused to save system resources? Do the features we have identified—including list-based and distance-biased random walk features—degrade as we consider ever more granular questions? We anticipate building on the large body of related work in general question-answer systems [Adamic et al. 2008; Mamykina et al. 2011; Wang et al. 2013a; Coetzee et al. 2014] to explore the factors impacting local expert-powered question-answer systems.

Query routing. For example, to avoid forwarding all questions to the top-ranked expert, we can use a round robin strategy or some other traffic scheduling scheme to divide and route user queries to different experts. We can also add features that characterize the time required for each expert to reply and incorporate this question urgency into how the query is routed.

Rare queries. In some cases, a rare query may not match any local expert at all. In these cases, we can default to a broader set of local experts by leveraging a user-generated knowledge base (e.g., a folksonomy built over local experts). For example, if there are few experts in "leveraged buyouts," we can relax the query to a broader category of "finance."

7. CONCLUSION

In this article, we proposed and evaluated a geo-spatial learning-to-rank framework for identifying local experts that leverages the fine-grained GPS coordinates of millions of Twitter users and carefully curated Twitter list data. We introduced five categories of features for the learning model, including a group of location-sensitive graph random walk features that captures both the dynamics of expertise propagation and physical distances. Through extensive experimental investigation, we find that the proposed learning framework that intelligently assigns feature weights can produce significant improvement compared to previous methods. By investigating the features, we found that high-quality local expert models can be built with fairly compact features. Additionally, we observed promising results for the reusability of learned models. In our future work, we seek to combine data from different social media platforms to better deal with sparsity. Furthermore, we are interested in investigating what new features may provide complementary evidence of local expertise.

REFERENCES

Lada A. Adamic, Jun Zhang, Eytan Bakshy, and Mark S. Ackerman. 2008. Knowledge sharing and yahoo answers: Everyone knows something. In *Proceedings of the 17th International Conference on World Wide Web*. DOI:http://dx.doi.org/10.1145/1367497.1367587

Akram Alkouz, Ernesto William De Luca, and Sahin Albayrak. 2011. Latent semantic social graph model for expert discovery in facebook. In *11th International Conference on Innovative Internet Community Services (I2CS 2011)*. GI Edition, 128–138.

Krisztian Balog, Leif Azzopardi, and Maarten De Rijke. 2006. Formal models for expert finding in enterprise corpora. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. DOI:http://dx.doi.org/10.1145/1148170.1148181

Krisztian Balog, Yi Fang, Maarten de Rijke, Pavel Serdyukov, and Luo Si. 2012. Expertise retrieval. *Foundations and Trends in Information Retrieval* 6, 2–3 (2012), 127–256. DOI:http://dx.doi.org/10.1561/1500000024

Mohamed Bouguessa, Benoît Dumoulin, and Shengrui Wang. 2008. Identifying authoritative actors in question-answering forums: The case of yahoo! answers. In *Proceedings of the 14th ACM SIGKDD*

International Conference on Knowledge Discovery and Data Mining. DOI:http://dx.doi.org/10.1145/1401890.1401994

Alessandro Bozzon, Marco Brambilla, Stefano Ceri, Matteo Silvestri, and Giuliano Vesci. 2013. Choosing the right crowd: Expert finding in social networks. In *Proceedings of the 16th International Conference on Extending Database Technology*. DOI:http://dx.doi.org/10.1145/2452376.2452451

Leo Breiman, Jerome Friedman, Charles J. Stone, and Richard A. Olshen. 1984. *Classification and Regression Trees*. CRC press. DOI:http://dx.doi.org/10.1002/widm.8

Christopher J. C. Burges, Krysta Marie Svore, Paul N. Bennett, Andrzej Pastusiak, and Qiang Wu. 2011. Learning to rank using an ensemble of lambda-gradient models. In *JMLR Workshop and Conference Proceedings: Yahoo! Learning to Rank Challenge*. Vol. 14. 25–35.

Christopher S. Campbell, Paul P. Maglio, Alex Cozzi, and Byron Dom. 2003. Expertise identification using email communications. In *Proceedings of the 12th International Conference on Information and Knowledge Management*. DOI:http://dx.doi.org/10.1145/956863.956965

Zhiyuan Cheng, James Caverlee, Himanshu Barthwal, and Vandana Bachani. 2014. Who is the Barbecue king of Texas? A geo-spatial approach to finding local experts on twitter. In *Proceedings of the 37th International ACM SIGIR Conference on Research and Development in Information Retrieval*. DOI:http://dx.doi.org/10.1145/2600428.2609580

Ed H. Chi. 2012. Who knows? Searching for expertise on the social web: Technical perspective. *Communication of the ACM* (April 2012). DOI:http://dx.doi.org/10.1145/2133806.2133829

Charles L. A. Clarke, Gordon V. Cormack, and Elizabeth A. Tudhope. 2000. Relevance ranking for one to three term queries. *Information Processing & Management* (2000). DOI:http://dx.doi.org/10.1016/S0306-4573(99)00017-5

Derrick Coetzee, Armando Fox, Marti A. Hearst, and Björn Hartmann. 2014. Should your MOOC forum use a reputation system? In *Proceedings of the 17th ACM Conference on Computer Supported Cooperative Work & Social Computing*. DOI:http://dx.doi.org/10.1145/2531602.2531657

Gao Cong, Christian S. Jensen, and Dingming Wu. 2009. Efficient retrieval of the top-k most relevant spatial web objects. *Proceedings of the VLDB Endowment* (2009). DOI:http://dx.doi.org/10.14778/1687627.1687666

Nick Craswell, Arjen P. de Vries, and Ian Soboroff. 2005. Overview of the TREC 2005 enterprise track. In *Text Retrieval Conference*. Vol. 5. 199–205.

Byron Dom, Iris Eiron, Alex Cozzi, and Yi Zhang. 2003. Graph-based ranking algorithms for e-mail expertise analysis. In *Proceedings of the 8th ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*. DOI:http://dx.doi.org/10.1145/882082.882093

Joseph L. Fleiss, Jacob Cohen, and B. S. Everitt. 1969. Large sample standard errors of kappa and weighted kappa. *Psychological Bulletin* (1969).

Saptarshi Ghosh, Naveen Sharma, Fabricio Benevenuto, Niloy Ganguly, and Krishna Gummadi. 2012. Cognos: Crowdsourcing search for topic experts in microblogs. In *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval*. DOI:http://dx.doi.org/10.1145/2348283.2348361

Jinwen Guo, Shengliang Xu, Shenghua Bao, and Yong Yu. 2008. Tapping on the potential of Q&A community by recommending answer providers. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management*. DOI:http://dx.doi.org/10.1145/1458082.1458204

L. I. Hang. 2011. A short introduction to learning to rank. *IEICE Transactions on Information and Systems* 94, 10 (2011), 1854–1862. DOI:http://dx.doi.org/10.1587/transinf.E94.D.1854

Gabriella Kazai, Jaap Kamps, Marijn Koolen, and Natasa Milic-Frayling. 2011. Crowdsourcing for book search evaluation: Impact of hit design on comparative system ranking. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*. DOI:http://dx.doi.org/10.1145/2009916.2009947

Jon M. Kleinberg. 1999. Authoritative sources in a hyperlinked environment. *Journal of the ACM (JACM)* 46, 5 (1999), 604–632. DOI:http://dx.doi.org/10.1145/324133.324140

Wen Li, Carsten Eickhoff, and Arjen P. de Vries. 2014. Geo-spatial domain expertise in microblogs. In *Advances in Information Retrieval*. Springer. DOI:http://dx.doi.org/10.1007/978-3-319-06028-6_46

Tie-Yan Liu. 2009. Learning to rank for information retrieval. *Foundations and Trends in Information Retrieval* (2009). DOI:http://dx.doi.org/10.1561/1500000016

Xiaoyong Liu, W. Bruce Croft, and Matthew Koll. 2005. Finding experts in community-based question-answering services. In *Proceedings of the 14th ACM International Conference on Information and Knowledge Management*. DOI:http://dx.doi.org/10.1145/1099554.1099644

Lena Mamykina, Bella Manoim, Manas Mittal, George Hripcsak, and Björn Hartmann. 2011. Design lessons from the fastest Q&A site in the west. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. DOI:http://dx.doi.org/10.1145/1978942.1979366

Catarina Moreira, Pável Calado, and Bruno Martins. 2011. Learning to rank for expert search in digital libraries of academic publications. In *Progress in Artificial Intelligence: 15th Portuguese Conference on Artificial Intelligence*. Springer. DOI:http://dx.doi.org/10.1007/978-3-642-24769-9_32

Wei Niu, Zhijiao Liu, and James Caverlee. 2016. LExL: A learning approach for local expert discovery on twitter. In *Proceedings of the 35th European Conference on Information Retrieval*. Springer. DOI:http://dx.doi.org/10.1007/978-3-319-30671-1_71

Aditya Pal and Scott Counts. 2011. Identifying topical authorities in microblogs. In *Proceedings of the Fourth ACM International Conference on Web Search and Data Mining*. DOI:http://dx.doi.org/10.1145/1935826.1935843

Aditya Pal, F. Maxwell Harper, and Joseph A. Konstan. 2012. Exploring question selection bias to identify experts and potential experts in community question answering. *ACM Transactions on Information Systems (TOIS)* 30, 2 (2012), 10:1–10:28. DOI:http://dx.doi.org/10.1145/2180868.2180872

Foster Provost and Pedro Domingos. 2003. Tree induction for probability-based ranking. *Machine Learning* 52, 3 (2003), 199–215. DOI:http://dx.doi.org/10.1023/A:1024099825458

Gang Wang, Konark Gill, Manish Mohanlal, Haitao Zheng, and Ben Y. Zhao. 2013a. Wisdom in the social crowd: An analysis of quora. In *Proceedings of the 22nd International Conference on World Wide Web*. DOI:http://dx.doi.org/10.1145/2488388.2488506

G. Alan Wang, Jian Jiao, Alan S. Abrahams, Weiguo Fan, and Zhongju Zhang. 2013b. ExpertRank: A topic-aware expert finding algorithm for online knowledge communities. *Decision Support Systems* 54, 3 (2013), 1442–1451. DOI:http://dx.doi.org/10.1016/j.dss.2012.12.020

Jianshu Weng, Ee-Peng Lim, Jing Jiang, and Qi He. 2010. Twitterrank: Finding topic-sensitive influential twitterers. In *Proceedings of the 3rd ACM International Conference on Web Search and Data Mining*. DOI:http://dx.doi.org/10.1145/1718487.1718520

Qiang Wu, Chris J. C. Burges, Krysta M. Svore, and Jianfeng Gao. 2008. Ranking, boosting, and model adaptation. *Tecnical Report, MSR-TR-2008-109* (2008).

Zi Yang, Jie Tang, and others. 2009. Expert2bole: From expert finding to bole search. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

Reyyan Yeniterzi and Jamie Callan. 2014. Constructing effective and efficient topic-specific authority networks for expert finding in social media. In *Proceedings of the 1st International Workshop on Social Media Retrieval and Analysis*. DOI:http://dx.doi.org/10.1145/2632188.2632208

Jun Zhang, Mark S. Ackerman, and Lada Adamic. 2007a. Expertise networks in online communities: Structure and algorithms. In *Proceedings of the 16th International Conference on World Wide Web*. DOI:http://dx.doi.org/10.1145/1242572.1242603

Jing Zhang, Jie Tang, and Juanzi Li. 2007b. Expert finding in a social network. In *Advances in Databases: Concepts, Systems and Applications*. Vol. 4443. Springer. DOI:http://dx.doi.org/10.1007/978-3-540-71703-4_106

Kathryn Zickuhr. 2013. Location-based services. *Pew Internet and American Life Project* (2013). http://www.pewinternet.org/files/old-media/Files/Reports/2013/PIP_Location-based%20services%202013.pdf.

Received December 2015; revised June 2016; accepted September 2016