

A LEARNING APPROACH FOR LOCAL EXPERT DISCOVERY

A Thesis

by

ZHIJIAO LIU

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of
MASTER OF SCIENCE

Chair of Committee, James Caverlee
Committee Members, Richard Furuta
Patrick Burkart
Head of Department, Dilma Da Silva

December 2015

Major Subject: Computer Science

Copyright 2015 Zhijiao Liu

ABSTRACT

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, this thesis: (i) proposes a geo-spatial learning-based framework, Local Expert Learning (LExL), for integrating multidimensional factors impacting local expertise, e.g. user-based, list-based, location-based and content-based features; (ii) accomplishes a comprehensive controlled study over AMT-labeled local experts on eight topics and in four cities, which not only leverages the candidates' basic information, but also considers the location authority impacting a candidate's expertise; and (iii) develops a prototype system, Local Experts Visualizing and Rating System (LEVRS), for visualizing and rating local experts. We find significant improvements (around 45% in precision and 50% in NDCG) of finding local experts compared to two state-of-the-art alternatives as well as evidence of the generalizability of the learned local expert ranking models to new topics and new locations.

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude and appreciation to my advisor, Professor James Caverlee, who has been a constant source of inspiration for me. Moreover, his excellent taste for important and novel research problems helped me become a more well-rounded person and build my own taste for research. Without his insights, advice, and extensive feedbacks, this work would have been impossible. Besides guiding my research, he provided a welcome and warmhearted lab environment, where every infolabber is the host of this big family. I am extremely fortunate to have him as an advisor.

I would also like to thank my thesis committee members, Professor Richard Furuta and Professor Patrick Burkart, for all of their guidance through this process. Their advice and feedbacks have been very helpful and invaluable to me.

To my lab-mates, all the other members of our Info Lab, especially Cheng Cao, Wei Niu, Haokai Lu, Hancheng Ge and Zhiyuan Cheng, thanks for the fun and support. They were always patient with any questions I would ask and glad to share their research experience and insights about my research with me.

Finally, I would like to dedicate this thesis to my parents, Shimin and Zhong, who have made many sacrifices to help me to pursue my dream and to be where I am today. For supporting me these years and for always being there for me no matter how hard time I faced, I thank you from the bottom of my heart.

Good, better, best. Never let it rest.

Until your good is better and your better is best.

TABLE OF CONTENTS

	Page
ABSTRACT	ii
ACKNOWLEDGEMENTS	iii
TABLE OF CONTENTS	iv
LIST OF FIGURES	vi
LIST OF TABLES	vii
1. INTRODUCTION	1
2. RELATED WORK	5
2.1 Expertise Retrieval	5
2.2 Work in Twitter-like Systems	5
2.3 Previous Local Expert Ranking Methods	6
3. LEXL: LOCAL EXPERT LEARNING	8
3.1 Problem Statement	8
3.2 Overview of Approach	9
3.2.1 “Hidden” Feature: Geo-Located Twitter Lists	9
3.2.2 Crawling Tweets for the Pool of Candidates	10
3.2.3 Learning Approach	11
4. FEATURES FOR LOCAL EXPERTISE	12
4.1 User-Based Features	14
4.2 List-Based Features	14
4.3 Local Authority Features	15
4.4 Tweet Content Features	17
5. EVALUATION	19
5.1 Experimental Setup	19
5.1.1 Queries and Candidates	19
5.1.2 Method: LExL	20

5.1.3	Baselines	21
5.2	Gathering Ground Truth	21
5.2.1	Pooling strategy	22
5.2.2	HIT design	22
5.2.3	Turker Agreement	24
5.2.4	Evaluation Metrics	25
5.3	Results	26
5.3.1	Comparison versus Baselines	26
5.3.2	Effectiveness Across Topics and Locations	28
5.3.3	Evaluating Feature Importance	30
5.3.4	Generalizability of Local Expert Models	34
6.	LEVRS: LOCAL EXPERTS VISUALIZING AND RATING SYSTEM	37
6.1	Design	37
6.2	Implementation	38
6.2.1	State view: Heatmap	39
6.2.2	Local view: Markers	40
7.	CONCLUSION AND NEXT STEPS	44
	REFERENCES	45

LIST OF FIGURES

FIGURE	Page
3.1 Twitter List Example	9
5.1 Design of HIT	23
6.1 Flowchart of LEVRS	38
6.2 State View and Local View of Local Experts	39
6.3 Basic Information of the Selected Expert	41
6.4 “Rate Me” Function	41
6.5 Set ranking range as a pre-filter	43

LIST OF TABLES

TABLE	Page
3.1 Geo-tagged Twitter list data	10
4.1 List of features used for ranking local expert candidates	13
5.1 Turker agreement for topics	24
5.2 Evaluating the proposed learning-based local expertise approach versus two alternatives. '†' marks statistical significant difference with LExL[LambdaMART] according to paired t-test at significance level 0.05.	27
5.3 Quality of local expert rankings across topics	29
5.4 Quality of local expert ranking in different locations	30
5.5 Quality of local expert ranking using different sets of features, '†' marks statistical significant difference with All Features case according to paired t-test at significance level 0.05.	31
5.6 Accumulated Times of Features Selected by Different Methods	32
5.7 Individual feature importance	33
5.8 Performance using selected features	34
5.9 Applying a model learned on one topic to rank local experts on a different topic.	36

1. INTRODUCTION

Identifying *experts* is a critical component for many important tasks. For example, the quality of movie recommenders can be improved by biasing the underlying models toward the opinions of experts [2]. Making sense of information streams – like the Facebook newsfeed and the Twitter stream – can be improved by focusing on content contributed by experts. Along these lines, companies like Google and Yelp are actively soliciting *expert reviewers* to improve the coverage and reliability of their services [11].

Indeed, there has been considerable effort toward expert finding and recommendation, e.g., [3, 6, 10, 16, 18, 21]. 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. Note that a local expert in one location may not be knowledgeable about a different location. To illustrate, consider the following two local experts:

- A “food” local expert in San Francisco is someone who may be good at cooking, very familiar with local restaurants or often post reviews or discounts of dishes in local restaurants.
- A “health and nutrition” local expert in Houston is someone who may be knowledgeable about local health providers, local health insurance options, local pharmacies and markets offering specialized nutritional supplements.

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 be applied in various aspects. For instance:

- **Surface content of local experts.** Currently, users mainly encounter Twitter content of other users they are following without regard for the expertise of these users. In many cases, a user may be interested in local content, so by exploiting local experts, we can instead surface content from these local experts focused on local information. For example, a traveler new to California can find local tasty dishes according to the tweets of food local experts in California.
- **Rerank Twitter stream.** Now tweets of a stream in Twitter are displayed in chronological order. With the detected local experts, the order of tweets can be reranked based on the impacts and emergency of a query topic in a specific location. Suppose at the time that Ebola case is found in Dallas, people in Dallas can obtain more local health precautions from the collection of health local experts in Dallas rather than the global report about Ebola.
- **Location-based recommendations.** Twitter presently provides Who-To-Follow feature to recommend other accounts that a user may have interests in. However, Who-To-Follow recommends the Twitter accounts basically according to the contacts of a user. If local experts can be recognized, then we can support location-based recommendations based on not only a user's current location, but also the query location in the user's search history, such as recommending "football" local experts in "Texas" where the two words have been searched for several times, or even show an advertisement of discount tickets for a Texas football game.

And yet, compared to general topic expert finding, there has been little research in uncovering local experts or on the factors impacting local expertise.

This thesis focuses on developing robust models of local expertise and visualization of the discovered local experts. More precisely, the main contributions of this thesis are:

- **Local Expert Learning (LExL) framework.** LExL is a geo-spatial learning-to-rank framework (Section 3) for identifying local experts, which makes use of the fine-grained GPS coordinates of millions of Twitter users, their relationships in Twitter lists and their tweet content is proposed and evaluated. The framework investigates multiple classes of features (Section 4) 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) list-based features (e.g., the number of lists the candidate is a member of, the number of lists the candidate has created);
 - (iii) local authority features (e.g., the distance between candidate and the query location, the average distance from a candidate’s labelers to the candidate);
 - (iv) tweet content features (e.g., the average tweet score calculated with TFIDF, the average tweet Entropy, etc.).

Through a controlled and comprehensive study (Section 5) over Amazon Mechanical Turk, the results of finding local experts with the proposed local expert learning approach have a large and significant improvement in multiple metrics, like Precision@10, NDCG@10, comparing to two state-of-the-art alternatives. In addition, the relative impacts of different groups of features are investigated, and the generalizability of the approach in order to reusing the learned models

is also examined. The findings indicate that the learning features can lead to finding local experts more accurately and high-quality local expert models can be built with fairly compact features, which shows potential adaptability of fitting more constrained cases. Finally, the proposed local expertise models are generalizable: in many scenarios, local experts can be discovered on new topics and in new locations with tradeoff of a little accuracy. This allows us to uncover unknown local experts in emerging areas.

- **Local Experts Visualizing and Rating System (LEVRS).** We pair the creation of LExL with a prototype local expert system called LEVRS. LEVRS embeds a LExL model and visualizes the learnt local experts (Section 6) on a map for showing the results we get from the experiments, which can also support future work on local experts study by incorporating the collection of new ground truth.

2. RELATED WORK

Previous work about local experts discovery can be classified into three mainly aspects: expertise retrieval, work in Twitter-like systems and previous local expert ranking methods.

2.1 Expertise Retrieval

Expertise retrieval has long been recognized as a key research challenge. The proposed methods can basically be separated into two categories according to the source of expertise indicators used. First, content-based methods leverage textual content and related documents which 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 QA systems, e.g., [17]. Al-Kouz et al. use user profile and post to match topic expertise on Facebook [1]. Balog et al. proposed a candidate generative model which represent a candidate directly by terms and a document model which first finds documents that are relevant to the topic and then locates the experts associated with these documents [3]. Second, graph-based methods based on social link analysis, so they consider each expert candidate's importance or social influence, e.g., [20]. For example, Campbell et al. utilize the link between authors and receivers of emails to improve expert finding in an email-based social network [6]. Moreover, there exist hybrid models considering both textual content and social relationships in expert finding, e.g., [18].

2.2 Work in Twitter-like Systems

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 [18]. Based on the list meta-data in Twitter, Ghosh et al. built the Cognos systems to help find experts on a specific topic [10]. 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 [7, 15]. Li et al. investigate expertise in terms of a user’s knowledge about a place or a class of places [15].

2.3 Previous Local Expert Ranking Methods

A previous approach by [7] focused on the local expert ranking problem using a linear combination of topical authority and local authority. In their work, Cheng et al. identified several factors, including local authority and topical authority, for assessing local expertise. 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 [3] 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 which 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.

Compared to these works, this thesis introduces the first learning-based method

for ranking local experts. 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 [12]. Thus, comparing the previous work on general expertise retrieval, Twitter-like System, and linear local expert ranking model, the learning-based method can utilize the collected four kinds of features to build a more general model and in the presence of ground truth training data to attack the problem of learning local experts.

3. LEXL: LOCAL EXPERT LEARNING

The learning approach framework for finding local experts – **LExL: Local Expert Learning** is introduced in this section. Given a query, containing a topic and a location, the goal of LExL is to identify high-quality local experts. To attack the problem of learning local experts, we need to formalize who can be defined as a local expert, and discuss how the learning approach can effectively solve this problem. The characteristics of the dataset and the reasons to choosing a specific learning to rank algorithm will also be discussed below.

3.1 Problem Statement

At first, we want to denote who can be defined as a local expert. The local experts we are looking for are people who are likely to live in or live close to the target local area, and are well recognized locally for their local knowledge about a particular topic. For example, Rudy’s, a Texas barbecue restaurant, may have higher local expertise for a specific query location, like College Station, than McDonald’s. Assume there is a pool of local expert candidates $V = \{v_1, v_2, \dots, v_n\}$, each candidate is described by a matrix of topic-location expertise scores (e.g., column i is College Station, while row j is “food”), and that each element of the matrix indicates the extent of expertise that the candidate is on the corresponding topic in the corresponding location. Given a query q that includes both a topic t and a location l , the 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 10 experts on $t_q = \text{“food”}$ in $l_q = \text{College Station, TX}$.

3.2 Overview of Approach

How to find local experts efficiently and accurately? To tackle this problem, a geo-spatial approach is proposed that integrates geo-crowd knowledge about each candidate into the learning-to-rank framework. Concretely, apart from the analysis of several obvious features of Twitter users, like the number of their followers and the text mining about the tweet content, the in-depth relationships in the geo-located Twitter lists has been exploited with LExL. Furthermore, we can find local experts faster by isolating the critical features that are more correlated with local expertise.

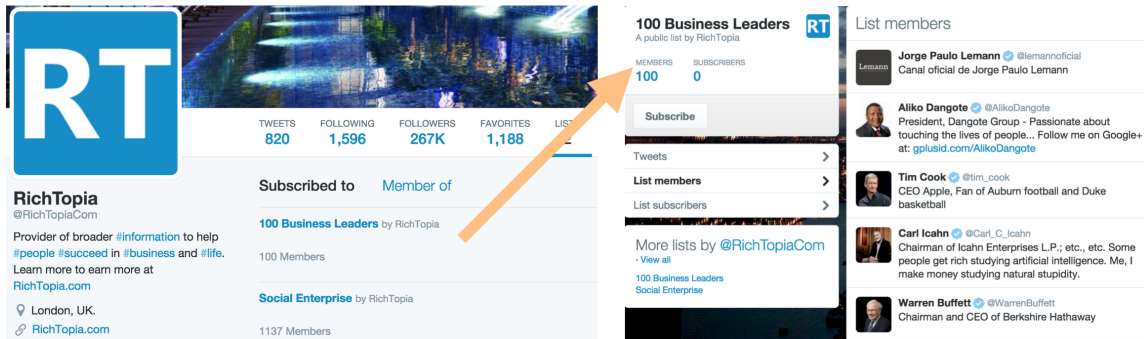


Figure 3.1: Twitter List Example

3.2.1 “Hidden” Feature: Geo-Located Twitter Lists

Besides the basic obvious features, like user profile and tweet content, are there any other features can reveal the subscribe relationships and topic expertise at the same time? A Twitter list allows a Twitter user to organize who she/he follows into logical lists. For example, Figure 3.1 shows one list named “100 Business Leaders” which contains 100 Twitter accounts including Tim Cook, Carl Icahn, and Warren Buffett. Twitter list is a form of crowd-sourced knowledge, which aggregates the individual judgement on a topic that the list represents for and can reveal the crowd

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

Table 3.1: Geo-tagged Twitter list data

perspective on how a Twitter user in the list is perceived [10]. In this thesis, the geo-social information of 13 million lists has been exploited – provided by the authors of [7]. The data provides the fine-grained location information of both the list creator (or *labeler*) and the member of the list (or *labeled*). In total, there are 86 million user occurrences on these lists, of which we have 15 million geo-tagged list relationships. Thus, the aggregate list information may reveal not only the *general* crowd perspectives on each user, but also the *local* crowd perspectives. High-level statistics of the dataset are listed in Table 3.1. Further details of the dataset collection method can be found in [7].

3.2.2 *Crawling Tweets for the Pool of Candidates*

What can determine a Twitter user has expertise in a specific topic? The first factor coming to our minds is the tweets they posted. The tweet content is the first evidence and foundation of their topical/locational expertise. For example, if a tweet is like “The new Kyle Field is awesome!”, then we can say this candidate has very high probability that she/he is in College Station and a fan or expert of football. So, we want to implement text mining to judge their level of expertise. One important step is crawling their tweets. After building the pool of local expert candidates according to the experimental setup (Section 5.1), for the 1387 filtered candidates in Twitter, the crawled tweets dataset is from May. 17, 2015 to Sep. 5, 2015, 16 weeks in total, which is processed to become the tweet content features

(Section 4.4).

3.2.3 Learning Approach

A previous approach by [7] focused on the local expert ranking problem using a linear combination of topical authority and local authority. To build a more general model and in the presence of ground truth training data (Section 5.2), here 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, which can combine any number of local expertise features.

To select the appropriate algorithm for learning, a library of learning to rank algorithms, RankLib,¹ is used since it currently supports eight popular algorithms. The following discussion mainly focuses on LambdaMART[5, 19], which has the best performance among four different learning to rank algorithms in the experiments (Section 5.3.1). The basic idea of LambdaMART is to train an ensemble of weak models and to linearly combine the prediction of each one of them into a final model which is stronger and more accurate. LambdaMART tunes the parameters of the regression trees based on a gradient-based optimization method, and the gradient of parameters is calculated according to the selected evaluation metric, e.g. NDCG, thus the evaluation metric is optimized directly when learning.

¹<http://sourceforge.net/p/lemur/wiki/RankLib/>

4. FEATURES FOR LOCAL EXPERTISE

How to represent expertise of a candidate accurately and comprehensively? In this part, four classes of features are introduced that potentially contribute to local topic expertise of a candidate: user-based features, list-based features, local authority features and tweet content features. This last two groups of features are especially important as they naturally integrate expertise propagation into a framework that models the relationships between different kinds of locations, and measure the relative level between tweet content and topic. All 33 features are summarized in Table 4.1.

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 that the user account was created on Twitter.
List-based Features	
6	N_{listed} The number of lists that this candidate appears on.
7	T_{listed} The number of on-topic lists that this candidate appears on.
8	N_{list} The number of lists this candidate has created.
9	T_{list} The number of on-topic lists this candidate has created.
10	S_{list} The average quality of the lists that the candidate is a member of.
Local Authority Features	
11	d_u Avg. distance from candidate c to all the users who appear on c 's lists.
12	d_{ut} Avg. distance from candidate c to all the users who appear on c 's on-topic lists.
13	d_l Avg. distance from candidate c to all the labelers of c .
14	d_{lt} Avg. distance from candidate c to all the labelers whose on-topic list has c .
15	d_{qt} Avg. distance from query location to labelers whose on-topic list has c .
16	d_{cq} Distance between candidate c and the query location l_q .
17	$Prox_c$ Candidate Proximity, as defined in Section 4.3.
18	$Prox_{sp}$ Spread-based Proximity, as defined in Section 4.3.
Tweet Content Features	
19	twP_c Avg. number of tweets that a candidate c posted in one week.
20	tw_s Avg. TFIDF score of a candidate c based on all the words in the posts.
21	tw_H Avg. tweet Entropy of a candidate c .
22-29	twB_t Topic bayesian scores of a candidate c in 8 topics, t .
30-33	twB_l Location bayesian scores of a candidate c in 4 topics, l .

Table 4.1: List of features used for ranking local expert candidates

4.1 User-Based Features

The first group of features captures user-oriented aspects that are independent of the query topic and query location.

- *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 that is both following and followed by the candidate.
- *User Activity* (N_{fav} , N_{status}): These two features are basic measures of a user’s activity-level on Twitter, where N_{fav} capturing 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 that an account was created. In this way, the longevity (or freshness) of the user can be integrated into the ranking model.

4.2 List-Based Features

The second group of features extract expertise evidence directly from the Twitter lists, but ignoring 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 [10]. 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, all lists as well as a more focused group of on-topic lists are considered (e.g., if the query is for “entrepreneurs”, we only consider entrepreneur-related lists, these lists are selected by keywords matching). Moreover, a new feature is defined to characterize the quality of on-topic lists. This new feature – list score, S_{list} – is defined as:

$$S_{list} = \frac{\sum_{i=1}^{N_{on_topic}(c)} Q_{list}(i)}{N_{on_topic}(c)}$$

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

$$Q_{list} = \frac{1}{k} \sum_{j=1}^k N_{on_topic}(j)$$

where k is the number of users in the list.

4.3 Local Authority Features

The third 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. The local authority of a candidate is measured in multiple ways:

- *Candidate-List Distance* (d_u, d_{ut}): 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. One version captures all of the lists (d_u) and one only considers on-topic lists (d_{ut}).

- *Candidate-Labeler Distance* (d_l, d_{lt}): The next two features measure the average distance from a candidate c to all of the users who have labeled c , capturing the localness of the people who have listed the candidate. Again, one version with all lists (d_l) and one with on-topic lists (d_{lt}) are considered.
- *Candidate-Query Distance* (d_{ql}, d_{cq}): These two features measure distance from the query location. The first (d_{ql}) is the 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, the distances are measured using the Haversine distance, which gives the great-circle distance around the earth’s surface. Apart from these six basic distance features, two features in a previous study of local experts [7] are also adopted: Candidate Proximity and Spread-Based Proximity. The Candidate Proximity is a decaying distance function between the candidate and a query location defined as:

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

where $d(l_c, l_q)$ denotes the Haversine distance between the candidate location l_c , 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:

$$Prox_{sp}(l(U_c), l_q) = \frac{\sum_{u \in U_c} Prox_c(l_u, l_q)}{|U_c|}$$

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” of a candidate is close to a query location on average, the candidate gets a high score of $Prox_{sp}$.

4.4 Tweet Content Features

As a key aspect of determining topical authority of a candidate, the tweet content cannot be omitted. Thus, the last group of features focuses on the tweet content, and it provides more statistical findings rather than raw data just based on counting.

- *Posting Frequency (twP_c)*: The average number of tweets that the candidate c posted in one week.
- *Tweet Score (tw_s)*: The average TFIDF per word of a candidate c . This feature can reflect how important and distinct the tweets that a candidate posted among all tweets.

$$tf-idf_{t,d} = tf_{t,d} \times idf_t$$

$$tw_s = \frac{\sum_{n_{week}} \sum_{t \in q} tf-idf_{t,d}}{n_t n_{week}}$$

$tf_{t,d}$ is the term frequency of a term t in one document (i.e. one week tweets of a candidate). idf_t is the inverse document frequency of a term t , as $\log \frac{N}{df_t}$, which

is composed by the total number of documents in a collection (one week tweets of all candidates), N , and document frequency, df_t , the number of documents in the collection that contain a term t .

- *Tweet Entropy (tw_H)*: The average Entropy of a candidate's one week tweets. This feature shows how informative the candidate's tweets are.

$$H = - \sum_{i=1}^{n_{word}} p(t_i) \cdot \log p(t_i)$$

$p(t_i)$ is the appearance times of a word in one document divided by the appearance times of all words in one document. tw_H is the average entropy according to each week's Entropy H .

- *Topic Bayesian Scores (twB_t)*: Use Naive Bayes method in scikit-learn library¹ to output the posteriors of all 8 topics for each candidate, which are calculated with priors learnt based on the topic pages crawled from Wikipedia.
- *Location Bayesian Scores (twB_l)*: Similar to Topic Bayesian Scores. Implement for 4 locations instead of the topics.

¹<http://scikit-learn.org>

5. EVALUATION

After introducing LExL, this section presents the experimental setup, including the collection of ground truth data via AMT, alternative local expert ranking methods, and metrics for comparing these methods. A series of experiments are designed to answer the following questions: What is the performance of this learning-based local expert ranking approach comparing to existing methods? Which features are most important for identifying local experts? How stable are the results across different topics and locations? Can a local expert model trained on one topic generalize to other topics? The results of these experiments show that LExL outperforms the alternatives and also has high generalizability.

5.1 Experimental Setup

The dataset used in the experiments is described in Section 3.2, which has 15 million geo-tagged list relationships in total.

5.1.1 *Queries and Candidates*

Our query set is built on a collection of eight topics and four locations similar to those in [7], which is representative and reflects real information needs. The topics are divided into general local expertise topics – “food”, “sports”, “business”, and “health” – and into more specialized local expertise topics – “chefs”, “football”, “entrepreneurs”, and “healthcare”. The locations are Chicago, Houston, New York City and San Francisco, which all have relatively dense coverage in the dataset for testing purposes.

We retrieved a set of candidates for ranking based on topics derived from list names before all experiments. Each list name has been applied case folding, stopword

removal, and noun singularization. String patterns like “FoodDrink” were separated into two tokens “food” and “drink”. Finally, each candidate is associated with all topics derived from this process, composing a set of potential candidates to be ranked by the proposed method LExL.

5.1.2 Method: LExL

There are a wide variety of learning-to-rank approaches accessible within an open source library, RankLib. In this thesis, four popular learning-to-rank strategies are evaluated: Ranknet, MART, Random Forest and LambdaMART. For each topic, the collected candidates are randomly partitioned into four equal-sized groups with their four categories of features for accomplishing four-fold cross validation to report the results. In LambdaMart, the NDCG value is optimized directly over the training set and averaged over all locations. Since LambdaMart adopts a boosting tree strategy (the number of boosting iterations is set to 1,000), the models can easily suffer from overfitting. Hence, we adopt three approaches to combat this overfitting. First, the model is tuned to have best NDCG value over a validation dataset that is partitioned from the training data. 1/3 of the training set is used for validation. Second, we adopt an early stopping strategy for terminating training once the NDCG does not improve. Third, we limit the number of trees and leaves per tree to 10 to prevent overfitting. Since LambdaMART performs the best in our evaluation and is significantly less computationally expensive (about 1/6 of the computing time of Random Forest), the following in-depth discussions are basically based on this method, and the discussion of the other three strategies, Ranknet, MART and Random Forest, is delayed to the experiments described below.

5.1.3 Baselines

Apart from the three strategies, LExL is also compared with two state of the art approaches as baselines for finding local experts:

- **Cognos+** [10]. 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, the Cognos implemented in our experiments is modified by incorporating a distance factor when calculating the cover density ranking [8], where each label is weighted by a distance factor range in $[0,1]$, which is similar to the way implemented in Candidate Proximity discussed in Section 4.3. Thus, this location-sensitive version of Cognos is referred as Cognos+.
- **LocalRank** [7]. The second baseline method is the LocalRank framework proposed in [7]. This framework ranks candidates by a linear combination of local authority and topical authority. The best performing combination reported in that paper, spatial proximity plus direct labeled expertise (SP+DLE), is chosen as the baseline to compare.

Note that both of these alternative methods are unsupervised, whereas the learning-based approach proposed here integrates labeled training data to bootstrap the ranker. Naturally, the supervised approach is expected to perform well. The goals here are to measure the improvement as well as investigate the importance of different features for local experts discovery.

5.2 Gathering Ground Truth

Since there is no publicly-available data that directly specifies a user’s local expertise given a query topic and a query location, 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).

5.2.1 Pooling strategy

It is too expensive to manually label the local expertise for 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 the effective utilization of turkers [14]. In order to build the pool of local expert candidates, the candidate set is sampled for each query pair, which only considers the candidates who appear at least one on on-topic list. 100 candidates are selected for each query pair, and then they are randomly assigned to different HITs.

5.2.2 HIT design

Each HIT includes instructions and examples of local expertise, Fig. 5.1a, along with twelve candidates to judge. Turkers can access the information about the query pair, Fig. 5.1b, 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 get familiar with the style of HITs with the task and make more consistent judgments. Several evaluation criteria [14] are adopted in order to collect high accuracy and reliable ground truth judgments. First, two out of the twelve candidate’s are set as trap questions, where we have already judged the candidates as either clearly local experts (4) or obviously having no local expertise (0). These

Please read this tutorial carefully before you start to rate the candidates.

Click [21812410](#) to see the candidate's profile.

Overview

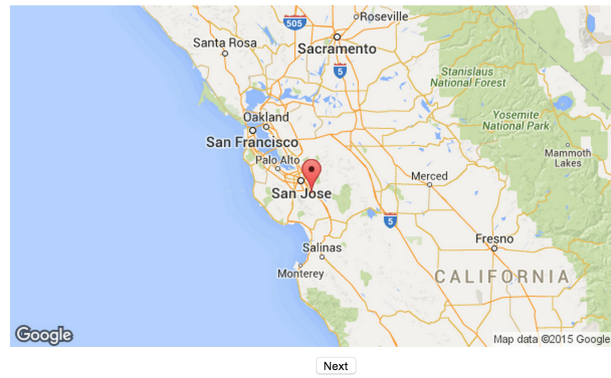
In this task, you will review 130 Twitter users as candidates of local experts, and help decide to what extent they have local expertise in $\$(query_topic)$ in $\$(location)$. You must have or [register a twitter account](#) in order to do the hit.

Task

For each candidate user, we ask you to help judge **to what extent the user has local expertise on the given topic in the given location**. The local experts we are looking for are [people who are likely to live in or live close to the target local area, and are well recognized locally for their local knowledge about a particular topic](#). And you are expected to select from "Extensive local expertise", "Some local expertise", "A little expertise", "No local expertise", or "No evidence" as the answer for each user's local expertise.

Your job is to look through the following information and decide the level of local expertise of the candidates.

- The candidate's user profile: including user's description, user's physical location, and user's personal URL.



Does candidate No.1 have business local expertise in San Francisco?

- 0 - No local expertise
- 1 - Difficult to tell
- 2 - A little local expertise
- 3 - Some local expertise
- 4 - Extensive local expertise

(a) Instructions for HIT

(b) Rating Questions

Figure 5.1: Design of HIT

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 candidates whose locations are Houston in 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 which only allows turkers whose results are consistently in 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.

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

Table 5.1: Turker agreement for topics

5.2.3 Turker Agreement

After run the HITs experiments, 16k judgments were collected in total across the eight topics and four locations based on the above settings. But are these assessments of local expertise reliable? To answer this, the *accuracy* and the *kappa statistic* [9] 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 inter-rater reliability, ranging from 0 to 1, with larger values indicating more consistency in the judgments. In Table 5.1, 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.

5.2.4 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 everyone 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’ judgements 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 that are actually local experts. Here, candidates with a rating 3 or 4 are considered as relevant; all others are irrelevant. Note that this is a more conservative approach than the one for inter-judge reliability; we want more distinguishing power deployed

between approaches for comparing local expertise methods.

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

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

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

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

$DCG@10 = \sum_{i=1}^{10} \frac{2^{rating_i-1}}{\log_2(i+1)}$ 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 , $rating'_i$ 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.

5.3 Results

In this section, the result of a series of experiments to evaluate the effectiveness of local expert finding using a learning-based method versus the two unsupervised methods and other in-depth study of the feature importance and model generalizability is reported as below.

5.3.1 Comparison versus Baselines

We begin with comparing the proposed learning method (LExL) versus the two baselines. Table 5.2 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.² First, we observe that three versions of LExL clearly outperform all alternatives, resulting in a Precision@10 of around 0.7, an average rating@10 of around 3, and an NDCG of more than 0.8.

Methods	Precision@10	Rating@10	NDCG@10
Cognos+	0.0906 [†]	1.456 [†]	0.2055 [†]
LocalRank	0.5049 [†]	2.491 [†]	0.5500 [†]
LExL [Ranknet]	0.6366 [†]	2.606 [†]	0.6846 [†]
LExL [MART]	0.6870	2.838	0.8353
LExL [Random Forest]	0.7247	3.026	0.8401
LExL [LambdaMART]	0.7137	2.922	0.8544

Table 5.2: Evaluating the proposed learning-based local expertise approach versus two alternatives. ‘†’ marks statistical significant difference with LExL[LambdaMART] according to paired t-test at significance level 0.05.

Cognos has been shown to be effective at identifying general topic experts. However, even a modified version, including 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 localizing these experts, which results in the poor performance.

LocalRank has a much better Precision@10 of around 0.5 compared to Cognos+,

¹Note that the results reported here for LocalRank differ from the results in [7] as the experimental setups and ground truth are different. First, our rating has 5 scales, which is intended to capture more detailed expertise level. Second, [7] only considers ideal ranking order for the top 10 results from LocalRank when calculating IDCG@10, while we consider a much larger corpus, thus the IDCG@10 is larger and this leads to smaller NDCG value.

²Ranknet is a pairwise learning method. For each pair of candidates (A,B), it aims to find the probability that candidate A has more local expertise than B. MART is a boosted tree model in which the output of the model is a linear combination of the outputs of a set of regression trees. Random Forest is a way of averaging multiple deep decision trees, trained on different parts of the same training set, with the goal of reducing the variance [13].

which indicates that 50 percent 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 it does not exploit either additional factors (like the tweet content features presented in this paper) nor take advantage of a learning approach for optimizing the weighting of these factors.

For the four LExL learning 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 (around 1/6 of the computing time of Random Forest), we focus our remaining discussion on this method.

5.3.2 *Effectiveness Across Topics and Locations*

Given the good performance of LExL with LambaMART, next we 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 5.3 that NDCG@10 is consistently high for the four general

topics, with an average value of 0.8276. 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, the learning framework is still effective at identifying even those few local experts in “health” since a high NDCG@10, 0.8674, it shows.

Topics	Precision@10	Rating@10	NDCG@10
food	0.7938	2.906	0.7288
sports	0.8625	3.444	0.8867
business	0.8125	3.306	0.8274
health	0.5562	2.273	0.8674
chefs	0.7750	2.925	0.8715
football	0.5953	2.634	0.9236
entrepreneurs	0.6889	2.767	0.7734
healthcare	0.6253	2.658	0.9565
General topic AVG	0.7563	3.098	0.8276
Subtopic AVG	0.6711	2.746	0.8813

Table 5.3: Quality of local expert rankings across topics

We observe comparable results for the four narrower topics. The Precision@10 is lower than for the general topics (0.67 versus 0.75), but the NDCG@10 is higher (0.88 versus 0.83). 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.

Locations	Precision@10	Rating@10	NDCG@10
Chicago	0.6901	2.758	0.8671
Houston	0.6410	2.601	0.8595
New York	0.6895	2.689	0.8625
San Francisco	0.6538	2.725	0.8694

Table 5.4: Quality of local expert ranking in different locations

In a similar fashion, the quality of LExL across the four query locations is evaluated as shown in Table 5.4. For the most part, the Precision@10, Rating@10, and especially NDCG@10 show good consistency across these four locations, suggesting the potential of a learning-based method to identify factors associated with each location for uncovering local experts.

5.3.3 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 upon four kinds of features – user-based, list-based, local authority, and tweet content features. 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 evaluate the quality of local experts identified.

From Table 5.5, the four 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), though not as well as the local authority features. These results suggest that intelligent combinations of many features via

Features	Precision@10	Rating@10	NDCG@10
User-based	0.6486 [†]	2.651 [†]	0.7439 [†]
List-based	0.6517 [†]	2.670 [†]	0.7713 [†]
Local Authority	0.6856 [†]	2.792	0.8366
Tweet Content	0.6404 [†]	2.593 [†]	0.7273 [†]
All Features	0.7137	2.922	0.8544

Table 5.5: Quality of local expert ranking using different sets of features, '†' marks statistical significant difference with All Features case according to paired t-test at significance level 0.05.

a learning method can outperform a simple combination of two carefully selected features (as in LocalRank). The Local Authority features achieve the highest Precision@10, Rating@10 and NDCG@10 among all four kinds of features. We attribute these results to Local Authority features integrating distance bias factors into capturing local expertise. We can observe that the combination of all features performs the best of all. One more interesting result is that Tweet Content features didn't perform as well as we expected. This can be explained by that the key words related to topic and location in tweet content didn't appear very often. For example, a NFL quarterback shares his feelings about life more often than football. Thus, we can say 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.³

- **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 required number of features are

³Both methods are provided in the Scikit Learn package: <http://scikit-learn.org/>

Feature	REF	Tree-based	Feature	REF	Tree-based
$N_{follower}$	2	5	twP_c	0	1
N_{friend}	1	2	tw_s	1	2
N_{fav}	1	1	tw_H	0	0
N_{status}	5	2	$twB_{business}$	0	0
T_{create}	5	1	$twB_{entrepreneur}$	0	0
N_{listed}	6	5	twB_{food}	0	0
T_{listed}	6	2	twB_{chef}	0	0
N_{list}	1	2	twB_{sport}	0	0
T_{list}	3	4	$twB_{football}$	0	1
S_{list}	7	0	twB_{health}	0	2
d_u	4	7	$twB_{healthcare}$	0	1
d_{ut}	6	5	twB_{chi}	0	0
d_l	0	3	twB_{hou}	0	1
d_{lt}	8	8	twB_{ny}	0	0
d_{ql}	8	8	twB_{sf}	0	1
d_{cq}	4	3			
$Prox_c$	8	8			
$Prox_{sp}$	4	5			

Table 5.6: Accumulated Times of Features Selected by Different Methods

left.

- Tree-Based Feature Selection.** In this approach, a number of randomized decision trees are built on various sub-samples 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 [4]. Features that attach to a node with higher *Gini importance* are more informative in the model.

Table 5.6 shows the accumulated number of times that each feature is selected by the two different feature selection methods. For 33 features, most of the top features are from list-based features and local authority features.

The results is aggregated across all queries (topics + locations), and reported

in Table 5.7 the top features for each feature importance method. There are seven common top features across both methods, which is highly consistent. Recall that d_{lt} and d_{ql} capture the average distance from candidate to all on-topic labelers and from query location to on-topic labelers respectively. $Prox_c$ shows the distance between the location of candidate and query location. Based on the selection results, we can say comparing to the tweet content, a candidate’s list and location-related features are more decisive for finding local experts.

Method	Top-10 Features
RFE	$d_{lt}, d_{ql}, Prox_c, S_{list}, N_{listed}, T_{listed}, d_{ut}, N_{status}, T_{create}, d_u$
Tree-based	$d_{lt}, d_{ql}, Prox_c, d_u, N_{listed}, d_{ut}, Prox_{sp}, N_{follower}, T_{list}, d_{cq}$

Table 5.7: Individual feature importance

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 5.8 shows the evaluation metrics for these two approaches versus the full blown LExL model with all features. We can observe the difference of Precision@10 and NDCG@10 is within 0.02 and Rating@10 is within about 0.1 for both methods compared to All Features case. Moreover, the difference between the value of three evaluation metrics and those of the All Features’ case is not statistically significant. These results confirm the importance of the List-based features and Local Authority features and further show that high-quality local expert models may be built using fairly compact features.

Method	Precision@10	Rating@10	NDCG@10
RFE	0.6965	2.805	0.8516
Tree-based	0.6919	2.797	0.8390
All Features	0.7137	2.922	0.8544

Table 5.8: Performance using selected features

5.3.4 Generalizability of Local Expert Models

Finally, can a learnt 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 5.9. For each of the four narrower topics, that indeed, the model corresponding to the most related general topic produces the best results. Perhaps surprisingly, these models perform on par

with the models trained over the individual topics as in Table 5.3, or even better in Precision@10 and Rating@10. And for the fluctuation of NDCG@10 is within 0.1 compared with that of their individual model. Since the general topic models build a more broader measure to define a local expert than the individual models of narrower topic, it can rank the more related candidates higher, which leads to high Precision@10 and Rating@10.

Even for models built on very different topics, we do see encouraging results. For example, the sports-based model for ranking chefs results in Precision@10 of 0.85, Rating@10 of 3.1, and NDCG@10 of 0.71 and the health-based model for ranking football results in Precision@10 of 0.79, Rating@10 of 3.1, 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. In Table 5.9 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 we may find a common local expert model that is applicable regardless of the specific topic.

Topic	Model	Precision@10	Rating@10	NDCG@10
healthcare	health	0.9250	3.708	0.9236
	food	0.6250	2.866	0.5927
	business	0.7937	3.350	0.8080
	sports	0.9250	3.708	0.8897
	general	0.8437	3.462	0.8285
entrepreneurs	health	0.7888	3.188	0.6867
	food	0.7111	2.755	0.5907
	business	0.8083	3.177	0.7663
	sports	0.7777	3.125	0.7060
	general	0.8000	3.066	0.6823
chefs	health	0.7333	2.941	0.6775
	food	0.8750	3.291	0.7670
	business	0.8687	3.225	0.7171
	sports	0.8583	3.175	0.7173
	general	0.8187	3.031	0.7132
football	health	0.7916	3.175	0.7910
	food	0.6166	2.683	0.6292
	business	0.7062	2.912	0.7157
	sports	0.8083	3.216	0.7965
	general	0.7312	2.956	0.7714

Table 5.9: Applying a model learned on one topic to rank local experts on a different topic.

6. LEVRS: LOCAL EXPERTS VISUALIZING AND RATING SYSTEM

The experiment results in previous section have shown LExL has satisfying performance on find local experts. Therefore, in this section, a prototype system LEVRS (Local Experts Visualizing and Rating System) is introduced where the system users can look for local experts with query of topic and location. Moreover, the system also receives user feedbacks for the displayed results, which can be used to adjust the proposed LExL.

6.1 Design

To design LEVRS, several questions are needed to take into account. What kind of data is it going to adopt? How to make the visualization clear and intuitive? How to make good use of the prototype system for future research? Thus, three key functions are considered, as follows.

- **Where are these local experts?** Since the research goal is finding local experts, it requires the system displays each detected local expert whose location is located on a real map. Moreover, we also displayed the distribution of different expertise levels across the country, for all candidate experts in our system.
- **Who are they?** Since our LExL is a ranking-based local expert detector, we should display the complete ranking and the basic profile information of each expert. Then system users can access the Twitter profile information of each local expert, including user name, profile photo, latest tweets, etc. Our LEVRS also illustrates the ranking of each expert and credit score he/she gained in LExL.

- **Not satisfied with the results?** As LEVRS shows the ranking of a selected expert, users may be not satisfied with the current results generated by LExL. To collect user feedback, we add a module extracting and displaying basic profile information of each found expert, and asking user to rate if they disagree.

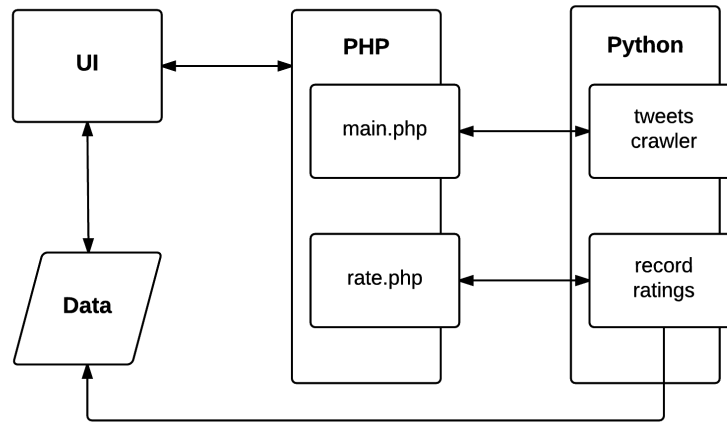
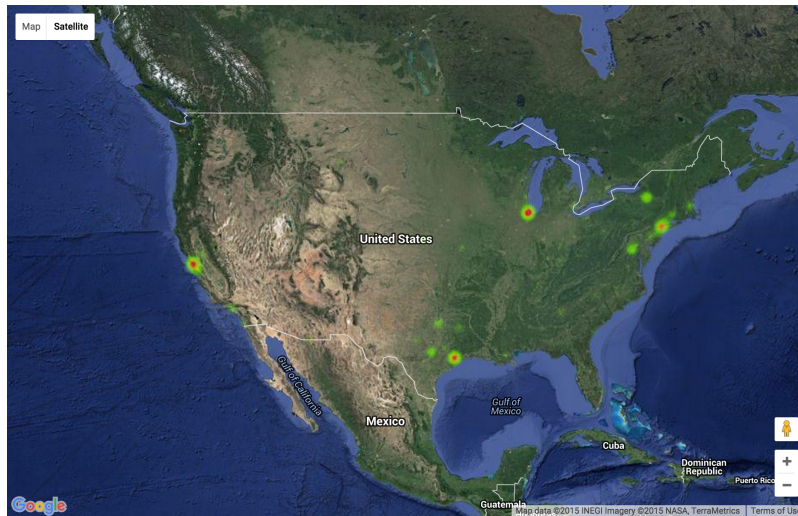


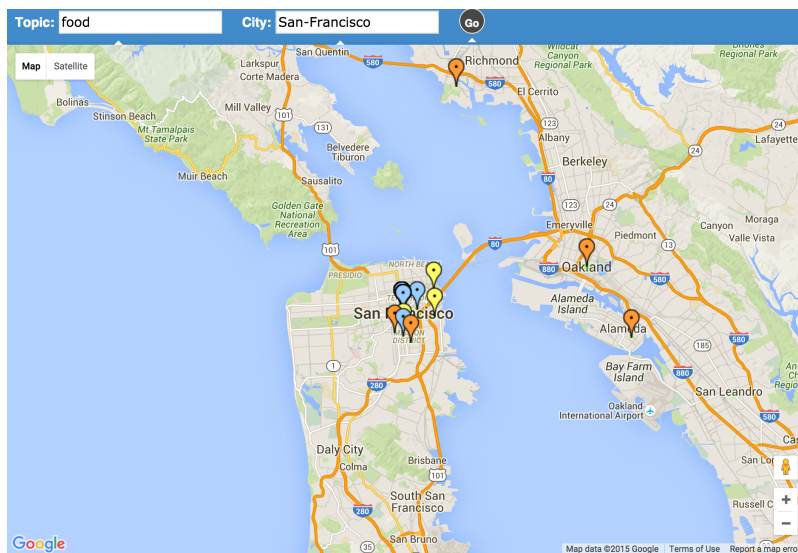
Figure 6.1: Flowchart of LEVRS

6.2 Implementation

As the requirements mentioned in Section 6.1, the prototype system, LEVRS, needs to support diverse views of local experts in a map, extraction of current information of an expert by interacting with Twitter API and ranking and rating data storage. Thus, basically the prototype LEVRS is implemented with 4 parts, User Interface, PHP layer, Python execution layer and Data IO. UI gets launched with the preprocessed data, like user locations, ranking info, etc. from Data IO. When users send requests to query local experts on one topic and in one location, UI will change from the state map view to a local view.



(a) Heatmap of Local Experts



(b) Markers of the Locations of Local Experts

Figure 6.2: State View and Local View of Local Experts

6.2.1 State view: Heatmap

Since the distribution of experts across the country may vary in different locations, a heat map is implemented in the state view. When opened from the main page of

LEVRS, it straightly shows the heat map of all the existing local experts across topics calculated by LExL . In Fig 6.2a, we can see there are more local experts in Chicago, Houston, New York and San Fransisco since the dataset only covers these 4 locations. But it also denotes that Houston and New York have relative sparse distributions comparing to Chicago and San Francisco. We attribute this to the population of San Francisco and Chicago are more concentrated in these two cities, while for New York and Houston, there are similar size cities close to them, for example, Washington DC and Dallas. Therefore, the candidates located in those cities are also being considered as local experts in New York and Houston respectively.

6.2.2 Local view: Markers

As individual, each local expert is required to be located on a real map by design. Hence, Fig 6.2b shows the exact locations of the local experts after LEVRS get the query of specific topic and location, for example, showing the food local experts in San Francisco. Additionally, different colors of the markers represent the rankings of these local experts in our prototype system, like the yellow markers are pinning the experts in the ranking range 50 to 100. Users can get a better understanding of their expertise level in the query area without knowing the exact rankings.

When one expert on the map has been selected, PHP layer will catch the request and execute related python crawler to extract current user profile information and latest 5 posts of this expert, shown in Fig. 6.3. Apart from the crawled information of an expert, the expert id links to the exact Twitter account of the expert, containing more types of profile informationsuch as number of followers, biography, and user-defined location. This provides the similar environment and process for rating a candidate. Therefore, a button, “Rate Me!” in Fig. 6.4, to present the rerating form is supported by UI in case users disagree with our rankings after they check the

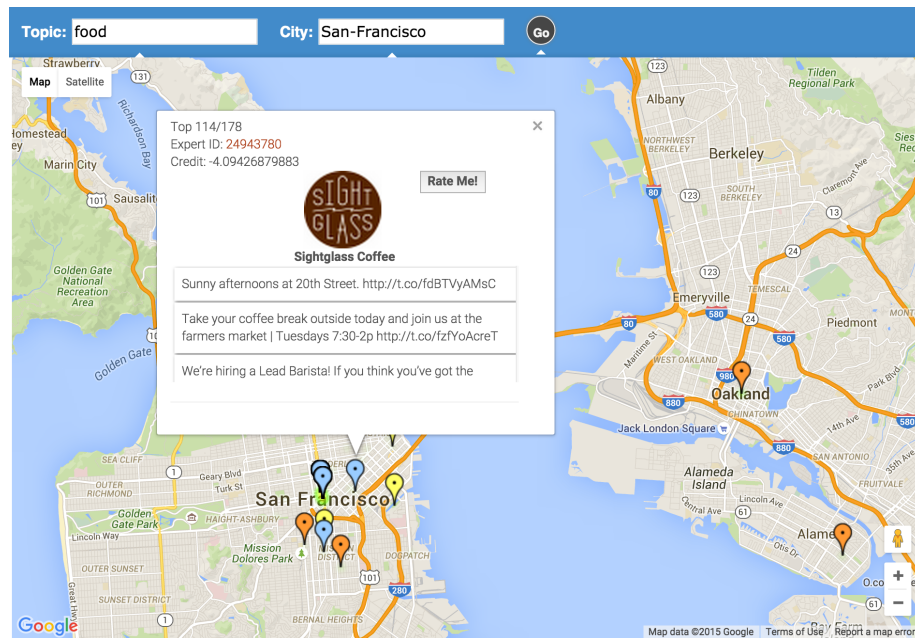



Figure 6.3: Basic Information of the Selected Expert

Top 49/178 ×
 Expert ID: **22997907**
 Credit: -1.80872511864


Bi-Rite Market

Escarole Salad-Cauliflower Pizza(GF!)-Ravioli-Pork Schnitzel-Lentil Salad #d

Friday! Come to @18reasons from 4:30-6:30 for our 75th Anniversary party! Cake, photo booth, music, fun, ALL the things!

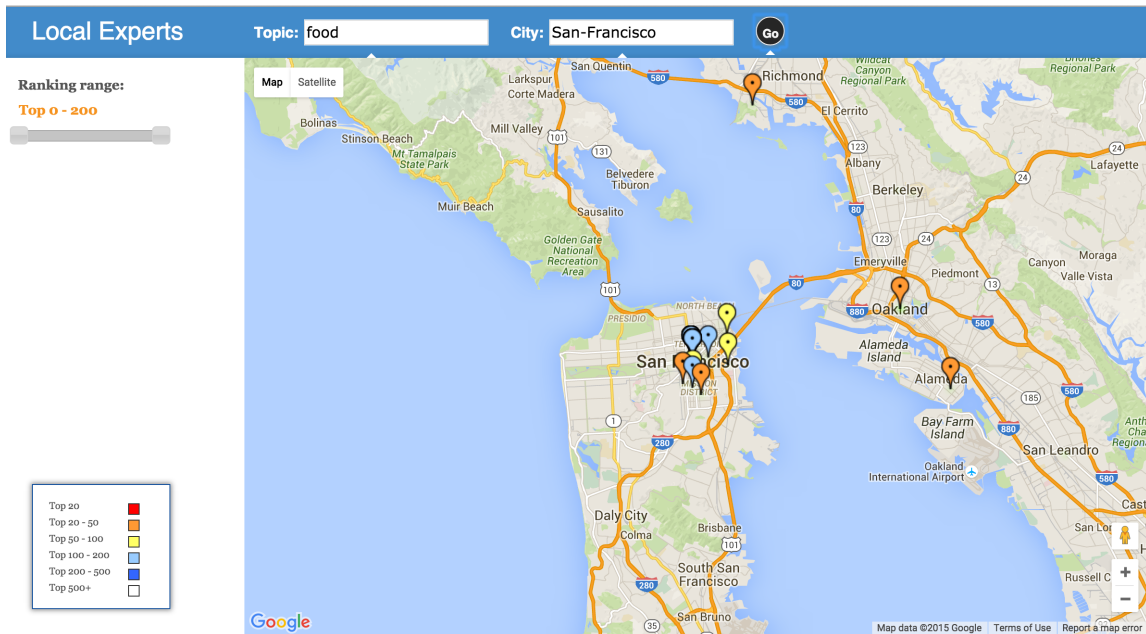
Does this user have **food** local expertise in **San-Francisco** ?

- 0 - No local expertise
- 1 - Difficult to tell
- 2 - A little local expertise
- 3 - Some local expertise
- 4 - Extensive local expertise

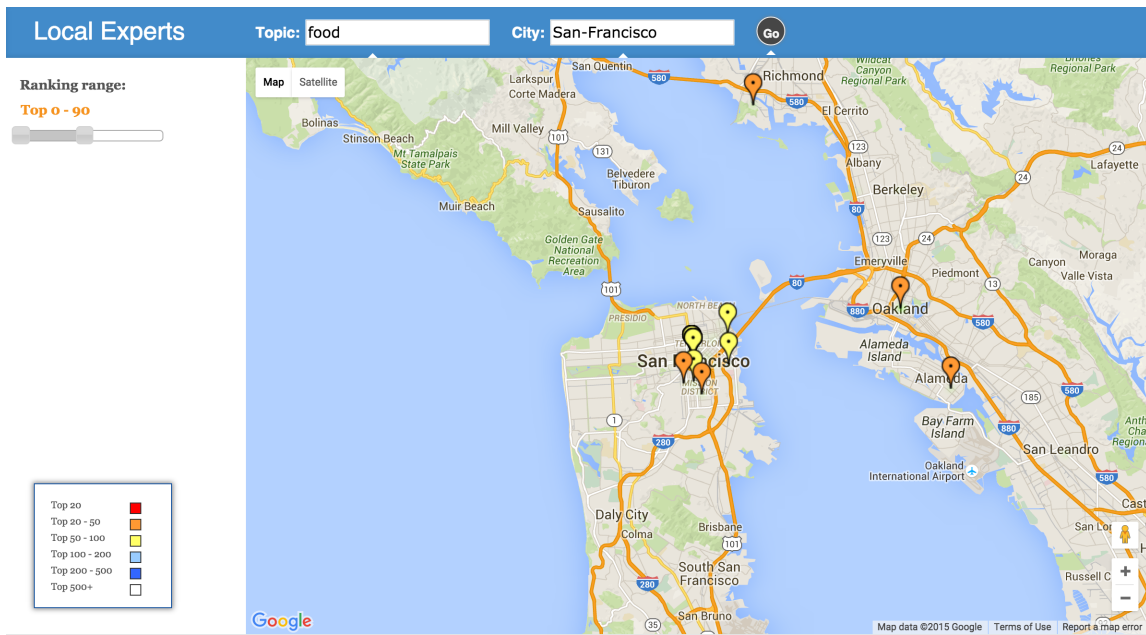
Figure 6.4: “Rate Me” Function

expert's current state. Furthermore, PHP layer and Python layer will store user ip, time of submission, rating score, expert id, respect topic and location with Data IO after the user submit the rerating form for adjusting the proposed LExL with these new ratings.

Seeing that for a query topic and location pair, LEVRS demonstrates the rankings of experts with distinct colors. However, it cannot display the distribution of experts with in a ranking range. Therefore, a ranking range filter is deployed to LEVRS so that system users can select the expected local experts within a ranking range. Fig 6.5 shows how ranking range bar works, and Fig 6.5b with range 0-90 is more easily showing that top 50 to top 100 local experts (yellow markers) are concentrated in San Francisco.



(a) Ranking Range is Top 200



(b) Ranking Range is Top 90

Figure 6.5: Set ranking range as a pre-filter

7. CONCLUSION AND NEXT STEPS

Local experts discovery plays a critical role to meet many location-sensitive information needs. In contrast to earlier lines of research aimed at finding general topic experts, we built and evaluated a geo-spatial learning-to-rank framework, LExL, for identifying local experts that leverages the fine-grained GPS coordinates of millions of Twitter user and carefully curated Twitter list data and tweet content data. Four categories of features for learning model are introduced, including user-based, list-based, local authority, and tweet content features. We accomplished a comprehensive controlled study over AMT-labeled local experts on eight topics and in four cities, where our proposed framework LExL performed better than alternatives. We developed LEVRS prototype system for visualizing and rating local experts.

The framework and prototype system we present thus suggest a number of directions for further work. (i) By discovering new group of features, we can potentially identify new relationships between topical and local authority of each local expert that can improve the performance of detected local experts. For example, now we only considered one-hop distance between a labeler and a candidate, so we can incorporate additional network context as new features. (ii) Currently, four learning-to-rank algorithms used in LExL have been proposed for many years. We can dive deep into the algorithms to publish a new one for boosting the performance. (iii) By enhancing the preliminary prototype system, we can provide real time rankings of the local experts with incorporating instant learning when some new ratings come into the system.

REFERENCES

- [1] Akram Alkouz, Ernesto William De Luca, and Sahin Albayrak. Latent semantic social graph model for expert discovery in facebook. In *IICS*, 2011.
- [2] Xavier Amatriain, Neal Lathia, Josep M. Pujol, Haewoon Kwak, and Nuria Oliver. The wisdom of the few: A collaborative filtering approach based on expert opinions from the web. In *SIGIR*, 2009.
- [3] Krisztian Balog, Leif Azzopardi, and Maarten De Rijke. Formal models for expert finding in enterprise corpora. In *SIGIR*, 2006.
- [4] Leo Breiman, Jerome Friedman, Charles J Stone, and Richard A Olshen. *Classification and regression trees*. CRC press, 1984.
- [5] Christopher JC Burges, Krysta Marie Svore, et al. Learning to rank using an ensemble of lambda-gradient models. In *Yahoo! LR Challenge*, 2011.
- [6] Christopher S. Campbell, Paul P. Maglio, Alex Cozzi, and Byron Dom. Expertise identification using email communications. In *CIKM*, 2003.
- [7] Zhiyuan Cheng, James Caverlee, Himanshu Barthwal, and Vandana Bachani. Who is the barbecue king of texas?: A geo-spatial approach to finding local experts on twitter. In *SIGIR*, 2014.
- [8] Charles LA Clarke, Gordon V Cormack, and Elizabeth A Tudhope. Relevance ranking for one to three term queries. *Information Processing & Management*, 2000.
- [9] Joseph L Fleiss et al. Large sample standard errors of kappa and weighted kappa. *Psychological Bulletin*, 1969.

- [10] Saptarshi Ghosh, Naveen Sharma, Fabricio Benevenuto, Niloy Ganguly, and Krishna Gummadi. Cognos: crowdsourcing search for topic experts in microblogs. In *SIGIR*, 2012.
- [11] Google. Overview of local guides, 2015.
- [12] LI Hang. A short introduction to learning to rank. *IEICE Transactions on Information and Systems*, 2011.
- [13] Trevor Hastie, Robert Tibshirani, Jerome Friedman, T Hastie, J Friedman, and R Tibshirani. *The elements of statistical learning*. Springer, 2009.
- [14] Gabriella Kazai, Jaap Kamps, Marijn Koolen, and Natasa Milic-Frayling. Crowdsourcing for book search evaluation: impact of hit design on comparative system ranking. In *SIGIR*, 2011.
- [15] Wen Li, Carsten Eickhoff, and Arjen P de Vries. Geo-spatial domain expertise in microblogs. In *Advances in Information Retrieval*. Springer, 2014.
- [16] Aditya Pal and Scott Counts. Identifying topical authorities in microblogs. In *WSDM*, 2011.
- [17] Aditya Pal, F. Maxwell Harper, and Joseph A. Konstan. Exploring question selection bias to identify experts and potential experts in community question answering. *ACM Transactions on Information Systems (TOIS)*, 2012.
- [18] Jianshu Weng, Ee-Peng Lim, Jing Jiang, and Qi He. Twiterrank: finding topic-sensitive influential twitterers. In *WSDM*, 2010.
- [19] Qiang Wu, Chris JC Burges, Krysta M Svore, and Jianfeng Gao. Ranking, boosting, and model adaptation. *Technical Report, MSR-TR-2008-109*, 2008.
- [20] Reyyan Yeniterzi and Jamie Callan. Constructing effective and efficient topic-specific authority networks for expert finding in social media. In *SoMeRA*, 2014.

- [21] Jun Zhang, Mark S Ackerman, and Lada Adamic. Expertise networks in online communities: structure and algorithms. In *WWW*, 2007.