Distributed Representations of Expertise

Fangqiu Han[*] Shulong Tan[*] Huan Sun[*] Mudhakar Srivatsa[†] Deng Cai[‡]
Xifeng Yan[*]

Abstract

Collaborative networks are common in real life, where domain experts work together to solve tasks issued by customers. How to model the proficiency of experts is critical for us to understand and optimize collaborative networks. Traditional expertise models, such as topic model based methods, cannot capture two aspects of human expertise simultaneously: Specialization (what area an expert is good at?) and Proficiency Level (to what degree?). In this paper, we propose new models to overcome this problem. We embed all historical task data in a lower dimension space and learn vector representations of expertise based on both solved and unsolved tasks.

Specifically, in our first model, we assume that each expert will only handle tasks whose difficulty level just matches his/her proficiency level, while experts in the second model accept tasks whose levels are equal to or lower than his/her proficiency level. Experiments on real world datasets show that both models outperform topic model based approaches and standard classifiers such as logistic regression and support vector machine in terms of prediction accuracy. The learnt vector representations can be used to compare expertise in a large organization and optimize expert allocation.

1 Introduction

Collaborative platforms, such as crowdsourcing service providers, community question answering forums, and customer service centers, are becoming more and more prevalent. Once managed effectively, the rich online human resources have shown great potential to solve problems more economically, efficiently, and reliably [1–3]. In order to effectively manage and utilize expert resources, an essential problem is how to correctly understand/represent human expertise and identify right experts for a certain task [4, 5]. In this paper, we take collaborative networks as an example to derive expertise representation so that multiple experts can be compared in the same framework.

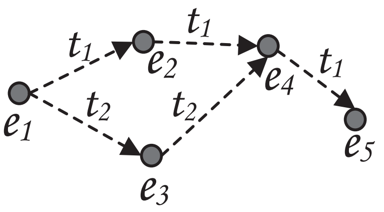

Figure 1: A sample collaborative network. Tasks are routed among experts in a collaborative network until they are resolved.

In collaborative networks, tasks are routed among a network of experts until they are resolved. Fig. 1 shows a sample collaborative network. Task t_1 starts at expert e_1 and is resolved by expert e_5; task t_2 starts at expert e_1 and is resolved by expert e_4. The sequences $e_1 \rightarrow e_2 \rightarrow e_4 \rightarrow e_5$ and $e_1 \rightarrow e_3 \rightarrow e_4$ are called *routing sequences* of task t_1 and t_2 respectively. One fundamental problem in ticket routing is how to represent experts' knowledge and employ it to estimate the probability of solving a task. Once this problem is solved, the final resolver to a given task can be found quickly.

An expert has to meet two constraints in order to solve a task: (1) Topic Match: the specialized areas of the expert shall match the topic of the task. For example, a programmer can possibly solve a programming problem while he is less likely to solve a physics problem; (2) Difficulty Level Match: the difficulty level of the task should match the proficiency of the expert. A programmer might be capable of implementing a *Binary Search* algorithm while he is unable to solve the *Eight Queen Puzzle*. These two constraints can be formalized as: (1) Specialization, i.e., the field an expert is specialized in; (2) Proficiency level: to what degree an expert is specialized in that field [6].

Previous studies [6, 7] assumed that a list of possible specialized areas for each expert is given. In real col-

[*]University of California, Santa Barbara. {fhan, shulongtan, huansun, xyan@cs.ucsb.edu.}

[†]IBM T.J. Watson Research Center. msrivats@us.ibm.com.

[‡]Zhejiang University. dengcai@gmail.com.

laborative networks, manually creating these specialized areas is laborious and hardly accurate. An intuitive solution is to use topic modeling such as Latent Dirichlet Allocation (LDA) [8] to automatically learn human expertise from the previously solved tasks. This solution has two main problems: (1) Tasks that an expert has failed to solve cannot be properly modeled to specify what the expert cannot do. (2) Topic models essentially capture the topic distribution of historical tasks. They do not directly measure proficiency level and its difference among experts.

To overcome the aforementioned issues and learn better expertise representations, we propose two expertise models in below:

(Model A) It assumes each expert has one or several specialized functional areas in a collaborative network. A task falling to one of the functional areas will be solved by the expert; otherwise it will be transferred to another expert. Based on this assumption, we define an expertise space in which all experts' expertise and all tasks will be embedded as numerical vectors. Tasks close to one of the expertise of an expert will be resolved by the expert whereas those far from his/her expertise will not. In this model, we combine the two aspects, specialized area and proficiency level of human expertise. The specialized area of an expert is characterized as a ball centered at his expertise vector and the radius of the ball signifies the range of the expert's duty. The ball is named *functional area* of the expert. This model is referred to as *Functional Area Expertise* (FAE).

(Model B) In some collaborative networks, there is no clear division of experts' responsibility. Experts solve tasks just based on their true capability. In this case, experts could deal with tasks in all difficulty levels below his capacity of solving tasks. Therefore, instead of unifying specialized areas and proficiency levels as in the FAE, our second model learns a vector representation of expertise and characterizes the two aspects separately: dimensions of the expertise vector encode specialized areas and the value in each dimension signifies the proficiency level of the expert on the corresponding area. Our intuitions in this formulation are as follows: (1) If an expert can solve a task, his proficiency level should be greater than or equal to the task difficulty; (2) If an expert cannot solve a task, there must be some dimensions in his expertise where their values are smaller than those required by the task. In this way, the specialized areas together with their proficiency levels can be modeled naturally. We refer to this model as *All-Round Expertise* (ARE).

FAE and ARE represent two different strategies of assigning task to experts. FAE is going to reserve the capacity of highly skilled experts for difficult tasks, while ARE tries to shorten task processing time as much as possible. We provide a comparison between FAE and ARE w.r.t several properties of collaborative networks, according to which one can decide the model that shall be used. Experimental results on real collaborative networks show that expertise learnt from our models better predict ticket solving than topic model based approaches and other methods in expertise modeling.

In comparison with previous studies on expertise modeling, to the best of our knowledge, we make the first attempt to consider all the factors together:

1. Not only do we utilize tasks solved by each expert but also those unsolved by him, which shall better characterize human expertise.

2. Not only do we characterize the specialized areas of an expert, but also the proficiency level he has in each area.

The rest of the paper is organized as follows. In Section 2, we briefly describe our problem setting. In Sections 3 and 4, the two proposed expertise models FAE and ARE will be introduced. Section 5 presents our experimental results. Related work is reviewed in Section 6, followed by the conclusion in Section 7.

2 Preliminaries

In this section, we introduce the notations and discuss the two aspects of expertise, i.e., specialized areas and proficiency level, that shall be captured by an expertise representation.

2.1 Task Routing and Resolution Records. In a collaborative network, a set of experts work cooperatively to solve tasks. Here we use $\mathcal{E} = \{E_i\}$ to represent the set of experts. Let $\mathcal{T} = \{t_j\}$ be a set of tasks resolved in the collaborative network, where each t_j is a bag-of-word vector with each dimension recording the word frequency in the task description.

Apart from the textual description, each task is also associated with a routing sequence starting from an initial expert to the final resolver of the task. Table 1 shows one example problem ticket in an IT service department. The ticket with ID 599 is a problem related to *operating system*, specifically, *the low percentage of the available file system space.* It was assigned to expert IN039, then routed through expert SAV59, and got resolved by expert SAV4F.

During task routing, an essential problem is to understand a certain expert's knowledge and estimate whether he could solve the current task or not. In this paper, we propose to automatically learn the expertise

Table 1: The Lifetime of A Task.

ID	Entry	Time	Expert
599	New ticket: the available space on the var file system is low	9/14/06 5:57:16	IN039
599	...(operations by IN039)...	...	IN039
599	Ticket 599 transferred to SAV59	...	IN039
599	...(operations by SAV59)...	...	SAV59
599	Ticket 599 transferred to SAV4F	...	SAV59
599	...(operations by SAV4F)...	...	SAV4F
599	Problem resolved: free up disk space in the file system	9/14/06 9:57:31	SAV4F

of all the experts based on the task routing and resolution records. Specifically, as shown in Table 1, the fact that expert IN039 and SAV59 did not solve task 599 but SAV4F solved it, will be leveraged to infer their expertise.

2.2 Expertise Representation. [6] gives a formal definition of expertise, which consists of two aspects: (1) Specialized areas of an expert; (2) Proficiency level of an expert in each specialized area. We formalize a distributed representation of expertise as follows:

$$< level(area_1), level(area_2), ..., level(area_d) >,$$

where $level(area_i)$ is the proficiency level of the expert in $area_i$ and d is the number of all possible specialized areas. Previous studies assume that all these specialized areas are pre-specified [6, 7]. In real collaborative networks, manually creating these specialized areas is often laborious and hardly accurate.

Here we introduce a novel distributed representation of expertise by embedding expertise and task in the same d dimensional space, referred to as the expertise space. We use e_i^k to represent the kth expertise of expert E_i. Each task t_j will be embedded in the expertise space by a transformation matrix W:

$$\tilde{t_j} = Wt_j, \tag{2.1}$$

where $\tilde{t_j}$ is the representation of the ith task t_j in the expertise space. Based on all the solved and unsolved tasks of each expert, we will learn their expertise representation e_i together with the transformation matrix W.

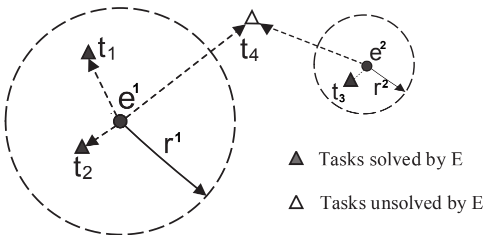

Figure 2: An intuitive example for Functional Area Expertise when $d = 2$. Tasks t_1 and t_2 are in the first functional area, t_3 is in the second functional area of E and can be solved by E. Tasks t_4 are out of both functional areas of expert E and thus cannot be solved by E.

3 Modeling Functional Area Expertise

In this section, we introduce our first expertise model called the Functional Area Expertise (FAE) model. In this model we assume each expert in the collaborative network may have one or more specialized functional areas and they could only solve tasks which belong to one of his functional areas. This assumption holds for many real collaborative networks in which explicit functional areas represent responsibilities.

Based on this assumption, we define a d dimensional Functional Area Expertise (FAE) space. Each expert will be assigned to one or several $d \times 1$ vectors, each represents the center of one functional area in expertise space and a corresponding radius parameter r which models the proficiency level in this area. In this new space, as shown in Fig. 2, tasks located within one of the functional areas of an expert E will be resolved by the expert and tasks located outside all the functional areas of the expert will not be resolved by E. An expert with larger radius parameter is likely to solve more tasks.

The functional area expertise model can be learned based on historical data. Intuitively, the learnt expertise should be close to his solved tasks but far from those tasks he cannot solve. We design the following objective function based on this intuition.

(3.2)
$$\begin{aligned}\underset{W,e,r_e}{\text{argmin}} &\sum_{E \text{ solved } t} \min_k f_{r_e^k}(||e^k - Wt||_2) \\ &- \sum_{E \text{ unsolved } t} \sum_k f_{r_e^k}(||e^k - Wt||_2) + \alpha||W||_1 \\ &+ \beta \sum_{e,1\le k_1,k_2\le k} \max(r_e^{k_1} + r_e^{k_2} - ||e^{k_1} - e^{k_2}||_2, 0),\end{aligned}$$

where r_e^k is the kth radius parameter for expert E, f is a monotonic increasing function referred later as the radius function, which will be described in detail later. The first term in the objective minimizes the distance between a solved task and solver's closest expertise center while the second term maximizes the distance between an unsolved task to expert's all expertise centers. The third term is a L_1 regularization term used to reduce model complexity and avoid overfitting. Intuitively different functional areas represent different expertise areas and thus should not overlap with each other. The last term in the objective gives penalty to functional area overlapping: it returns 0 when the distance between any two expertise centers is greater than the sum of two corresponding radius parameters and returns a positive penalty otherwise.

Simply setting function f as the identical function leads to a trivial solution. For example if expert E fails to solve task t, the objective function returns negative infinity when we embed all expert centers of E to origin and ticket t to infinity. To avoid this, we set f to be the shifted sigmoid function defined as follows:

$$f_r(x) = \frac{1}{1 + \exp -(x - r)},$$

where r is the radius parameter.

We applied well-established L-BFGS (Limited-memory BFGS) algorithm [9] to optimize objective function Eq. (3.2). L-BFGS is an optimization algorithm in the family of quasi-Newton methods. Instead of storing a dense approximation of the inverse Hessian matrix, L-BFGS algorithm stores only a few vectors that represent the approximation implicitly. This makes it particularly well suited for optimization problems with a large number of variables. Based on learnt W, r_e, and e, the probability of an expert E solving a task t can be predicted by:

$$1 - \min_k f_{r_e^k}(||e^k - Wt||_2).$$

4 Modeling All-Round Expertise

In some collaborative networks, the assumption above for the functional area formulation may not hold. Instead, an expert can solve tasks whose difficulty level is equal to or below his/her proficiency level. In this section, we propose an All-Round Expertise (ARE) model to handle this scenario. ARE is based on the following intuition: (1) If an expert E can solve a task t, its expertise should be greater than or equal to t in all dimensions. (2) If the expert cannot solve the task, its expertise should be smaller than the task at least in some dimensions. An example with $d = 2$ is shown in Fig. 3. e_1, (a, b), and e_2, (c, d), are expertise of two experts. Tasks lying inside the smaller rectangular can be solved by expert E_1. All tasks lying inside the bigger rectangular can be solved by expert E_2. In this model, dimensions are considered as areas and the value on each

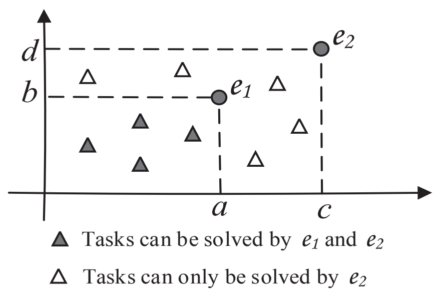
d
b
e2
e1
a
c
▲ Tasks can be solved by e1 and e2
△ Tasks can only be solved by e2

Figure 3: An example for the All-Round Expertise model. Expert E_1 can only solve the tasks covered by the small rectangular. But E_2 can solve all the tasks including those E_1 can solve.

dimension is the proficiency level of an expert in that area. Therefore, we no longer need to assign several expertise vectors to one expert.

4.1 ARE for Solved Tasks. Let $e = (e_{(1)}, e_{(2)}, \ldots, e_{(d)})$ be the expertise of expert E and $\tilde{t} = (\tilde{t}_{(1)}, \tilde{t}_{(2)}, \ldots, \tilde{t}_{(d)})$ be the vector representation of task t in the expertise space. We define a new operation $\dot{-}$ in the expertise space as follow:

$$e \dot{-} \tilde{t} = \sum_{1 \le l \le d} (\min(0, e_{(l)} - \tilde{t}_{(l)})).$$

Note that $e \dot{-} \tilde{t}$ is always smaller than or equal to 0 and $e \dot{-} \tilde{t} = 0$ if and only if $e_{(l)} \ge \tilde{t}_{(l)}, \forall k$. We define the objective function of ARE for solved tasks is as follows:

$$\operatorname*{argmax}_{W,e} \sum_{e \text{ solved } t} (e \dot{-} \tilde{t}). \tag{4.3}$$

We denote dimension k as a **strong dimension** for expert E with respect to task t if $e_{(l)} \ge \tilde{t}_{(l)}$, and as a **weak dimension** with respect to t if $e_{(l)} < \tilde{t}_{(l)}$. When E solved t, the objective function in Eq. 4.3 penalizes all weak dimensions and do not care how 'strong' the strong dimensions are. If all the dimensions are strong, we have $e \dot{-} \tilde{t} = 0$, which maximizes $e \dot{-} \tilde{t}$. The motivation is, if the expert can solve the task, this expert shall have knowledge deep enough on all the areas required by t.

Only learning experts' expertise from the solved tasks will not work well. For example in order to tell one task t_j is more difficult than another task t_i, we need information like some expert E solved t_i but not t_j. This can also be illustrated by a trivial solution of the above model which embeds all the tasks to 0 and all expertise to some positive number. This solution clearly maximizes the above objective but fails to learn the difficult level for each task. Base on this intuition, we shall include unsolved tasks in learning expertise.

4.2 ARE for Unsolved Tasks. For unsolved tasks, we believe there must be some weak dimensions for expert E. For the optimization purpose, we set a margin parameter $\alpha \geq 0$ to measure how weak these dimensions are. To be specific, if expert E fails to solve task t, the sum of differences between expertise e and t over all weak dimensions should be smaller than a negative threshold, which is set to be $-\alpha$. Fig. 4 shows an example of this margin with $d = 2$. If an expert E could solve task t, the learnt expertise e should be located in *area* 1, while if e fails to solve t, the learnt expertise representation should be located in *area* 2. We use following formula to measure the penalty for unsolved tasks:

$$\min(-\alpha - (e \dot{-} \tilde{t}), 0),$$

where $e \dot{-} \tilde{t}$ can be considered as the sum of differences between e and t over all weak dimensions. If the sum is larger than $-\alpha$, the formula equals to $-\alpha - (e \dot{-} \tilde{t})$, which is negative. Otherwise, the formula returns 0.

Margin α distinguishes between solved tickets and unsolved tickets of an expert in the expertise space. Setting margin α to 0 leads to a trivial solution: the objective function returns a global minimum value 0 for both solved and unsolved tasks when mapping all expertise and ticket vectors to the origin. We will describe in detail how to select α later in the experiment section.

We define the objective function of ARE for unsolved tasks as follows:

$$(4.4) \qquad \underset{W,e}{\operatorname{argmax}} \sum_{e \text{ unsolved } t} (\min(-\alpha - (e \dot{-} \tilde{t}), 0)).$$

When E fails to solve t, this objective function penalizes all dimensions when $e \dot{-} \tilde{t}$ is above the margin.

4.3 Objective Function of ARE. We now combine penalties from both solved and unsolved tasks to the objective function:

$$(4.5) \; \underset{W,e}{\operatorname{argmin}} \; - \sum_{E \text{ solved } t} (e \dot{-} \tilde{t}) - \sum_{E \text{ unsolved } t} (\min(-\alpha - (e \dot{-} \tilde{t}), 0)) + \beta ||W||_1.$$

Similar to the FAE model, we add an additional L_1 regularization term and apply L-BFGS algorithm to optimize ARE. Denote the objective function in Eq. 4.5

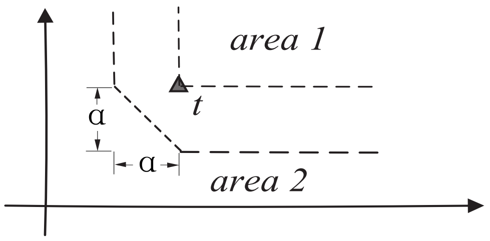

Figure 4: An example of applying margin α in the All-Round Expertise model with $d = 2$. The expertise of all experts who are capable of solving t is expected to be located in *area* 1, while the expertise of all experts who cannot solve t is expected to be located in *area* 2.

as $\mathcal{H}(W, e)$. The derivative of $\mathcal{H}$ with respect to $\{W, e\}$ is given by:

$$(4.6) \; \begin{aligned} \frac{\partial \mathcal{H}}{\partial W_{ij}} &= \sum_{E \text{ solved } t} t_{(j)} * I(e_{(i)} < \tilde{t}_{(i)}) \\ &\quad - \sum_{E \text{ unsolved } t} t_{(j)} * I(E \dot{-} \tilde{t} > -\alpha) \\ &\quad + \beta * sgn(w_{ij}), \\ \frac{\partial \mathcal{H}}{\partial e_{(i)}} &= \sum_{E \text{ solved } t} -1 * I(e_{(i)} < \tilde{t}_{(i)}) \\ &\quad + \sum_{E \text{ unsolved } t} 1 * I(e \dot{-} \tilde{t} > -\alpha), \end{aligned}$$

where $sgn(x)$ is the sign function and $I(x)$ is the indicator function which returns 1 if statement x is true and returns 0 otherwise. Based on W and e learnt from ARE, whether an expert can or cannot solve a task can be predicted by $e - \tilde{t}$. The decision boundary is determined using 5-fold cross validation on training. The scale of e's each dimension indicates how good the expert is in the corresponding area. Similarly, the scale of t's each dimension indicates how difficult the task is in that dimension.

5 Experiments

In this section, we evaluate the proposed two expertise models, FAE and ARE, on real-life datasets. We test their performance in predicting whether an expert E can solve a task t.

5.1 Baselines. The performance of FAE and ARE is compared with standard classifiers and other popular methods in expertise modeling.

(1) **Logistic Regression (LR).** We train a logistic regression model [10] for each expert. This model takes a bag-of-word vector as input and outputs the

probability that this expert solves the task. This probability, denoted as $P_i(t)$, is defined as follows:

$$P_i(t) = \frac{1}{1 + exp(-(W_i * t + b_i))},$$

where W_i is the weight vector associated with expert E_i which has the same size as the bag-of-word vector of tasks. W_i can be viewed as an expertise vector learnt for each expert; $W_i * t$ computes the dot product similarity between E_i and t, which is used to predict task solving. The parameters in the model are learned by minimizing the square-loss error, based on the $< expert, task >$ pairs in the training dataset.

$$\underset{W,b}{\text{argmin}} = \sum_{<e_i,t>} (P_i(t) - y_{<E_i,t>})^2,$$

where y is 1 if E_i solved t and is 0 otherwise.

(2) **SVM.** We build an SVM classifier for each expert and use it to predict whether he/she can solve the tasks in the testing set. We tried different kernels including linear, polynomial, quadratic and multilayer perceptron; their best results are reported.

(3) **Query Likelihood Language Model (QLL).** In QLL [11], each document is represented as a multinomial distribution over words. The maximum likelihood estimate of this distribution is the frequency of each word in the document divided by the total number of words in the document. We apply Dirichlet smoothing to the distribution as most of the words in the vocabulary do not show together in each individual document. The likelihood of a query task t generated from a document d under a language model with Dirichlet smoothing is defined as:

$$p(t|d) = \prod_{w \in t} \frac{N_d}{N_d + \mu} p(w|d) + \frac{\mu}{N_d + \mu} p(w),$$

$$p(w|d) = \frac{N_d(w)}{N_d},$$

where N_d is the number of words in d, $N_d(w)$ is the number of word w in d, μ is a smoothing parameter, and $p(w)$ is the probability of the word w in the entire corpus. For each expert E we construct two document collections, C_e^+, which is a collection of all his/her solved tasks, and C_e^-, which is a collection of all his/her unsolved tasks. Intuitively, for an expert E, if a new task is close to one task in the solved task collection, then this new task is highly likely to be solved by E. Similar argument works for the unsolved task collection. Thus we define the likelihood of a query task t obtained from a document collection C as:

$$p(t|C) = \max_{d \in C} p(t|d).$$

For expert E and a new task t, we predict E solve t if $p(t|C_e^+) > p(t|C_e^-)$ and E cannot solve t otherwise.

(4) **Topic Modeling.** Topic modeling can be used to learn expertise. Specifically, we first create two documents for each expert by merging task descriptions in C_e^+ and C_e^-, and denote them as d_e^+ and d_e^- respectively. Latent Dirichlet Allocation (LDA) [8] is then used to train topic models. For each expert, two topic distributions θ_e^+ and θ_e^- are learned corresponding to d_e^+ and d_e^-. θ_e^+ conveys what expert E can do and θ_e^- tells what he/she cannot do. The likelihood of a task t generated from a topic distribution is defined as:

$$p(t|\theta_e) = \prod_{w \in t} \sum_{z \in Z} \theta_e^z \times p(w|z),$$

where z is one of Z topics and $p(w|z)$ is the word probability under topic z which is output by the topic model. For expert E and a new task t, we predict E solve t if $p(t|\theta_e^+) > p(t|\theta_e^-)$ and e cannot solve t, otherwise. Advanced topic models, such as [12], are not chosen as baselines because they do not fit our problem setting. For example, there is no multiple authors for a task as a scientific paper does. Since each task description in our dataset typically contain a few words, directly applying complex topic models will not work well due to sparse word co-occurrence [13].

5.2 Datasets. We use real-world problem ticket data collected from a problem ticketing system in an IT service department throughout 2006. Two datasets in different problem categories are explored: Windows and AIX, which contain problem tickets occurring in the Windows and AIX operating systems. The details of both datasets are shown in Table 2. The data is quite sparse: Only a few tasks were recorded for each expert and there are a few words in each task description.

For each dataset we only keep experts who have solved and unsolved at least 3 tasks.

Table 2: Two Datasets on Task Resolution

Datasets	# of tasks	# of experts
WIN	32349	1278
AIX	10519	1988

We apply well-established L-BFGS algorithm to optimize the objective function in FAE, ARE, and LR. All parameters are initialized randomly from $[-1, 1]$ and updated iteratively using the second order gradients estimated by L-BGFS. We conduct a $80\% - 20\%$ random split on the dataset and generate training and testing

Table 3: Accuracy on task resolving prediction

Models	WIN(%)	AIX(%)
FAE	85.09 ± 0.49	85.38 ± 1.06
ARE	86.52 ± 0.62	88.18 ± 0.63
LR	81.44 ± 0.65	79.48 ± 0.78
SVM	80.14 ± 0.40	79.17 ± 0.59
QLL	78.44 ± 0.47	67.55 ± 3.92
LDA	65.60 ± 0.67	74.20 ± 0.49

data. 5-fold cross validation is used on the training dataset for hyper parameters selection on all the methods including FAE and ARE, e.g., α, β in Eq. 3.2 and β in Eq. 4.5. The margin parameter α in ARE should not effect the expertise learning: if W and e are local minimal for Eq. 4.5 with α then $2W$ and $2e$ are local minimal for Eq. 4.5 with 2α. We use the 5-fold cross validation to determine the decision boundary for both FAE and ARE. The whole process is repeated for 5 times; the mean and standard deviation of the classification accuracy over the 5 runs are reported. The bag-of-word representation uses the top $2,000$ most frequent words in all the tickets as used in [14]. The dimensionality of expertise is set at 20 for both FAE and ARE models. In the FAE model, each expert is assumed to have 2 specialized areas (k is set at 2 empirically).

5.3 Accuracy. Table 3 summarizes the performance of all the methods. As shown, the proposed two models work significantly better than all the baselines in terms of prediction accuracy. As the ticket dataset is sparse, LR, SVM and QLL cannot learn very good classifiers as they learn a classifier for each individual expert. Our models learn expertise for all the experts together and embed all the tasks in the same expertise space. The learnt transformation matrix W is shared by all the tasks and the sparsity issue is overcome in this way. The performance of our models is better than these three baselines. Not surprisingly, the topic model method, LDA, works badly in this case. The reason is topic models highly rely on word co-occur information. Tasks in our data are very short (the average number of words in a task is around 5) and only a few short documents are associated with each expert. Therefore, topic modeling cannot learn proper topic distributions. ARE outperforms FAE significantly in the AIX dataset indicating that the WIN and AIX collaborative networks might adopt different task assignment strategies.

5.4 Efficiency. The time complexity of the FAE and ARE models is determined by the number of gradient update iterations multiplied by the complexity of computing the objective derivatives in each iteration. As shown in Eq. 4.6, the complexity of the derivatives computation is $O((d_t + N) * d)$, where d_t is the dimensionality of ticket bag-of-word representation, d is the dimensionality of the expertise space, and N is the number of ticket-expert pairs. We did not show the derivative of the objective function in FAE due to the space limit. However, it is easy to see that the complexity is the same as ARE. Note that both models may take a long time to converge. Here we set a limit on iterations and stop the model training after 50 iterations. Figure 5 shows how we empirically select this number by plotting the classification accuracy versus the number of training iterations for both models on the AIX dataset.

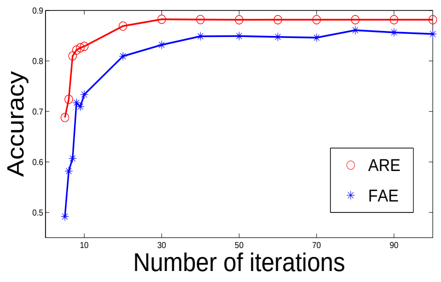

Figure 5: Classification accuracy versus the number of iterations for FAE and ARE on the AIX dataset. Both models almost converge after 50 iterations.

Table 4: Efficiency on model learning

Models	WIN	AIX
FAE	$5'21''$	$3'38''$
ARE	$5'20''$	$3'02''$
LR	$27''$	$28''$
SVM	$3'23''$	$7'9''$
QLL	$1'3''$	$2'31''$
LDA	$11'59''$	$18'23''$

Table 4 summarizes the running time for each model. We list the training time for all the methods. These algorithms are implemented using MATLAB on a 8-core 3.40GHz Intel CPU with 16G memory. The testing time for all the methods is shorter than 1 second, which means after expertise learning all the methods can do prediction in a timely manner.

6 Related Work

In this section, we first introduce the background of collaborative platforms and then review previous methods on expertise modeling and representation.

Collaborative Platforms. Recent years have witnessed blossoms of various collaborative platforms, such as community question answering forums Quora[1] and Stack Overflow[2]. Collaborative network is a typical example of collaborative platforms, where a task is routed through the network until being resolved. Studies have been focused on developing automated routing algorithms to route a task to its final resolver as fast as possible [15–17]. Miao et al. [18] studied a network model and a routing model to jointly simulate the structure and the task routing procedure in a collaborative network. Sun et al. [14] studied the routing behaviors of experts in collaborative networks and modeled their decision making process on where to transfer a task. In this paper, we study a key problem in collaborative networks: expertise modeling and representation, which serves the central role in many applications, such as expert search and expertise comparison.

Expert Search. Expert search aims at finding experts who are knowledgable on a given topic and capable of solving a given problem [19–22]. Expert search became an important research area since it started at the TREC enterprise track [19] in 2005. The crucial problem in expert search is how to rank experts given a query. Many papers in this line consider expertise modeling as a part of expert ranking. No explicit expertise representations are learned [23, 24]. For example, Deng et al. formalized expert ranking by statistical language model and topic-based model [24]. Fang et al. [22] proposed a discriminative learning framework to directly model the conditional probability of relevance between an expert and a query. In this paper, we try to learn a vector representation for each expert, which can be used to compare expertise in a large organization and optimize expert allocation.

Expertise Modeling. In different application scenarios, various types of data are used to model expertise, such as project descriptions and professional articles [11, 20–22]. For example, Mimno et al. [11] modeled a reviewer's expertise by the papers she has written when matching reviewers with submitted papers. Guan et al. [25] mined the fine-grained knowledge of a Web user, by analyzing their Web surfing data, to facilitate expert search in collaborative environments. In question answering, Zhang et al. [5] used language models to compute user expertise based on the threads a user contributes to, where each thread contains a question post and a number of reply posts. Li et al. [26] incorporated question category into question routing, where question category is used to estimate answerers' expertise. Chang et al. [27] proposed a routing framework that uses compatibility, availability and expertise of the users to recommend answerers and commenters to a question. Expertise is modeled by using questions an expert has answered and questions' tags annotated by askers.

As social media becomes prevalent, studies (e.g., [4, 28, 29]) on using social media data to derive user expertise, have emerged recently. Yeniterzi et al. [28] employed authority signals such as votes, comments, and follower/following information to estimate user expertise. In [4], Guy et al. provided an extensive study that explores the use of social media to infer user expertise, by evaluating the data users produced through a large survey. Varshney et al. [29] inferred the expertise of employees in the IBM corporation through mining social data that is not specifically generated and collected for expertise inference. Our work on modeling expertise distinguishes itself from these studies: We not only take into account tasks that are solved by an expert, but also those unsolved by her, in order to characterize her expertise.

Expertise Representation. For expertise representation, early approaches built a knowledge base which contained the descriptions of people's skills within an organization [30] or used tags to represent expertise of each expert [31]. More advanced methods are based on topic modeling [11,12]. Topic Modeling is a standard approach to explain the observed data. Latent Dirichlet Allocation (LDA) [8], which takes a set of documents as input and simultaneously learn the document-topic and topic-word distributions. For each expert, one can combine the set of tasks solved as a single document. LDA takes the documents of all the experts and outputs the topic distribution for each expert as his/her expertise representation. However, topic distributions learnt by LDA essentially capture experts' historical task distributions but not their true capability on task solving, such as the proficiency level on a specialized area. In contrast, expertise learnt by our models conveys both aspects of human expertise: specialization and proficiency level.

7 Conclusion

Expertise modeling is considered as the core content in improving the efficiency of collaborative networks. The goal is to represent experts' abilities in terms of specialization and proficiency level. In this paper, we developed two models to learn distributed representations of expertise which can convey both aspects. The shared insight of these two models is embedding both expertise and tasks that were solved/unsolved by experts in

[1]http://www.quora.com/
[2]http://stackoverflow.com/

the same space. The embeddings are optimized over all historical data points: which expert solved which task and which expert failed to solve which task. The first model assumes that an expert only solves tasks matching his/her specialization and proficiency level. Alternatively, in the second model, the expert can solve all the tasks whose difficulty level is equal to or lower than his/her proficiency level in his/her specialized area. Experimental results on real datasets demonstrated that the proposed models can learn meaningful expertise representations and are effective in predicting task resolution.

References

[1] P. Welinder, S. Branson, P. Perona, and S. J. Belongie, "The multidimensional wisdom of crowds," in *NIPS*, 2010, pp. 2424–2432.

[2] D. R. Karger, S. Oh, and D. Shah, "Efficient crowdsourcing for multi-class labeling," in *ACM SIGMETRICS*, vol. 41, no. 1, 2013, pp. 81–92.

[3] S. Jagabathula, L. Subramanian, and A. Venkataraman, "Reputation-based worker filtering in crowdsourcing," in *NIPS*, 2014, pp. 2492–2500.

[4] I. Guy, U. Avraham, D. Carmel, S. Ur, M. Jacovi, and I. Ronen, "Mining expertise and interests from social media," in *WWW*, 2013, pp. 515–526.

[5] Y. Zhou, G. Cong, B. Cui, C. S. Jensen, and J. Yao, "Routing questions to the right users in online communities," in *ICDE*, 2009, pp. 700–711.

[6] K. Balog and M. De Rijke, "Determining expert profiles (with an application to expert finding)," in *Proc. of IJCAI-07*, 2007, pp. 2657–2662.

[7] K. Balog, Y. Fang, M. de Rijke, P. Serdyukov, and L. Si, "Expertise retrieval," *Foundations and Trends in Information Retrieval*, vol. 6, no. 2–3, pp. 127–256, 2012.

[8] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *JMLR*, vol. 3, pp. 993–1022, 2003.

[9] D. C. Liu and J. Nocedal, "On the limited memory bfgs method for large scale optimization," *Mathematical programming*, vol. 45, no. 1-3, pp. 503–528, 1989.

[10] C. Bishop, "Pattern recognition and machine learning," 2007.

[11] D. Mimno and A. McCallum, "Expertise modeling for matching papers with reviewers," in *SIGKDD*, 2007, pp. 500–509.

[12] M. Rosen-Zvi, C. Chemudugunta, T. Griffiths, P. Smyth, and M. Steyvers, "Learning author-topic models from text corpora," *TOIS*, vol. 28, no. 1, p. 4, 2010.

[13] X. Yan, J. Guo, Y. Lan, and X. Cheng, "A biterm topic model for short texts," in *WWW*, 2013, pp. 1445–1456.

[14] H. Sun, M. Srivatsa, S. Tan, Y. Li, L. M. Kaplan, S. Tao, and X. Yan, "Analyzing expert behaviors in collaborative networks," in *SIGKDD*, 2014, pp. 1486–1495.

[15] Q. Shao, Y. Chen, S. Tao, X. Yan, and N. Anerousis, "Efficient ticket routing by resolution sequence mining," in *SIGKD*, 2008, pp. 605–613.

[16] G. Miao, L. E. Moser, X. Yan, S. Tao, Y. Chen, and N. Anerousis, "Generative models for ticket resolution in expert networks," in *SIGKDD*, 2010, pp. 733–742.

[17] H. Zhang, E. Horvitz, Y. Chen, and D. C. Parkes, "Task routing for prediction tasks," in *AAMS-Volume 2*, 2012, pp. 889–896.

[18] G. Miao, S. Tao, W. Cheng, R. Moulic, L. E. Moser, D. Lo, and X. Yan, "Understanding task-driven information flow in collaborative networks," in *WWW*, 2012, pp. 849–858.

[19] N. Craswell, A. P. de Vries, and I. Soboroff, "Overview of the trec 2005 enterprise track," in *TREC*, vol. 5, 2005, pp. 199–205.

[20] K. Balog, L. Azzopardi, and M. De Rijke, "Formal models for expert finding in enterprise corpora," in *SIGIR*, 2006, pp. 43–50.

[21] P. Serdyukov, H. Rode, and D. Hiemstra, "Modeling multi-step relevance propagation for expert finding," in *CIKM*, 2008, pp. 1133–1142.

[22] Y. Fang, L. Si, and A. P. Mathur, "Discriminative models of integrating document evidence and document-candidate associations for expert search," in *SIGIR*, 2010, pp. 683–690.

[23] X. Liu, W. B. Croft, and M. Koll, "Finding experts in community-based question-answering services," in *CIKM*, 2005, pp. 315–316.

[24] H. Deng, I. King, and M. R. Lyu, "Formal models for expert finding on dblp bibliography data," in *ICDM*, 2008, pp. 163–172.

[25] Z. Guan, S. Yang, H. Sun, M. Srivatsa, and X. Yan, "Fine-grained knowledge sharing in collaborative environments," *TKDE*, 2015.

[26] B. Li, I. King, and M. R. Lyu, "Question routing in community question answering: putting category in its place," in *CIKM*, 2011, pp. 2041–2044.

[27] S. Chang and A. Pal, "Routing questions for collaborative answering in community question answering," in *ASONAM*, 2013, pp. 494–501.

[28] R. Yeniterzi, "Effective approaches to retrieving and using expertise in social media," in *SIGIR*, 2013, pp. 1150–1150.

[29] K. R. Varshney, V. Chenthamarakshan, S. W. Fancher, J. Wang, D. Fang, and A. Mojsilović, "Predicting employee expertise for talent management in the enterprise," in *SIGKDD*, 2014, pp. 1729–1738.

[30] T. H. Davenport and L. Prusak, *Working knowledge: How organizations manage what they know.* Harvard Business Press, 1998.

[31] A. John and D. Seligmann, "Collaborative tagging and expertise in the enterprise," in *WWW*, 2006.