2009 International Conference on Computational Science and Engineering

Expertise Modeling and Recommendation in Online Question and Answer Forums

Suratna Budalakoti, David DeAngelis, and K. Suzanne Barber
Laboratory for Intelligent Processes and Systems
The University of Texas at Austin
1 University Station C5000, Austin, TX 78712, USA
Email: {sbudalakoti, dave, barber}@lips.utexas.edu

***Abstract*—Question and answer forums provide a method of connecting users and resources that can leverage both the static and dynamic (live) capabilities of a network of human users. We present a recommender for selecting the most appropriate responders given a question. The goal of this work is to encourage expert participation in QA forums by reducing the time investment needed by an expert to find a suitable question, decrease responder load, and to increase questioner confidence in the responses of others. The two primary contributions of this work are: 1. a generative model for characterizing the production of content in an online question and answer forum and 2. a decision theoretic framework for recommending expert participants while maintaining questioner satisfaction and distributing responder load. We have also developed two new metrics tailored to QA forums: responder load and questioner satisfaction. These metrics are used to evaluate the performance of our recommender system on datasets harvested from Yahoo! Answers. Experiments across several topic domains demonstrate our systems ability to predict responder identities and suggest new responders that are likely to have the appropriate expertise.**

I. Introduction

Untapped capabilities permeate large-scale networks. Search engines specialize in identifying existing static documents on a network that are appropriate for a given query. Question and answer (QA) forums provide a method of connecting users and resources that can leverage both the static and dynamic (live) capabilities of a network of human users. No single user has complete knowledge across many different domains. On a large network, however, it is likely that somebody has expertise in nearly every question domain. We have developed a framework that facilitates the flow of information from those that have it to those that need it. An implementation of this framework will allow users to answer questions that benefit from uniquely human insight. These types of questions will often be recommendation-based, and built on qualitative tradeoffs that are best suited for human judgment. For such a system to be successful, it is essential that it be able to identify and access experts in any given area, and connect the responder(s) to the original questioner. This research presents an algorithmic approach to fulfill these aims.

An effective QA system must facilitate the sharing of information. A question is provided by a user called a questioner and any other user (called a responder) is capable of reading this question and providing a response. In this work we propose that a question is supplied, and the QA system must select potential responders from which to solicit responses. This selection process, or responder recommendation, is the core of our QA framework. Recommendation is a challenging and popular problem in the data mining and machine learning communities, to the extent that Netflix offers a US$1M prize for an improved movie recommender [4].

Identification of expertise is the first step in recommending a responder. Expertise is defined as the ability of a user to answer a given question to the satisfaction of the questioner. Given a question, how can the pool of potential answerers be indexed and searched to predict who is capable of and willing to provide an answer. Estimating which user is most likely to give a satisfactory answer is a challenging problem, and requires a complex model of human expertise along: a) expertise dimensions: the various distinct areas of human knowledge, and the experts ability in each of these areas, b) compatibility: the likelihood that the answerers personality and approach to answering questions matches that of the questioner, c) willingness: the probability that the answerer will be willing to invest the time required to answer the question.

Multi-dimensional models of expertise are crucial to the problem, as an expert on one subject is not necessarily an expert on another, even when the two areas are closely related. While question-answer systems already exist on the web such as Yahoo! Answers and the defunct Google Answers, in most current implementations, there is little or no modeling of user expertise, and hence an expert is expected to wade through many questions until finding one that is most suitable. Very specialized questions may never be viewed by the few qualified experts, and the result is that only simple and generic questions receive an answer. By automating the process of finding an expert for a question, we remove the investment of time required by an expert to find a suitable question, thereby reducing the cost of participation and improve overall productivity.

II. Contributions

There are two main contributions of this work:

1) We propose to model expertise in a Q&A system using higher level concepts we call *expertise topics*, which are associated with distributions over words as well as experts. These distributions are used to calculate

978-0-7695-3823-5/09 $26.00 © 2009 IEEE
DOI 10.1109/CSE.2009.62

the probability of an author having relevant expertise given a question, and these probabilities are used to make a recommendation according to our decision-theory framework. This method captures the historical word usage as well as participation patterns for users. We propose a finite mixture based generative model to discover topics, and estimate the parameters of the model using the expectation-maximization (EM) algorithm.

2) We also propose a decision theoretic framework for recommending expert participants, that tries to find satisfactory responses for all questions without overloading any expert with too many questions.

An alternative to our generative model is to use a standard clustering algorithm such as the k-means algorithm, and identify these clusters as topics. An even simpler alternative is to forgo the explicit definition of expertise topics. This is accomplished by associating words with authors based on historical usage and then recommending author responders based on the similarity between the words in the question and the responders word usage. We intend to show that author-topic and word-topic models for describing Q&A behavior lead to better responder recommendations than a traditional scheme of clustering words and authors into topics or simple author-word counts.

Our recommender currently adopts the questioner perspective and recommends the most appropriate expert responders to answer a given question. It is possible to take this same recommendation mechanism to recommend a question of the appropriate topic to an idle responder. This is reserved for future work.

A second experiment demonstrates the effectiveness of applying our decision-theoretic framework to the responder recommendation problem. Without the framework, it is possible to simply assign a question to a topic and choose the most appropriate author according to the author-topic distributions. We call this the *best match* technique. We hypothesize that this will lead to undesirable system properties such as the overloading of the most qualified experts with too many questions. The decision-theoretic framework is designed to balance the responder load and the satisfaction of the questioners, and our experiment will investigate its performance according to system-wide metrics.

III. Related Work

Peer production is a term used to describe the phenomenon of distributed users collaborating to contribute value to others without oversight or management by a business enterprise. In peer production systems users, or agents, are motivated by three types of rewards: monetary, intrinsic hedonic, and social-psychological [3]. Many instances of peer production forgo monetary rewards completely, such as Wikipedia and the series of Games with a Purpose [14]. These games intrinsically reward volunteer users for performing tasks such as image tagging. Not all peer production systems are entirely benevolent; Pouwelse et al. explain that many systems contain Pirates and Samaritans [10]. Pirates may add value by illegally sharing content at the expense of the content creators. Question and answer forums are another form of peer production, where volunteer users add value by answering questions posed by others. It is possible for a pirate to intentionally supply low quality answers, but most mechanisms driving QA forums do not reward this behavior.

One popular example of a QA forum is Yahoo! Answers (YA). Whereas Google Answers provided a monetary incentive to answer questions, Yahoo! Answers relies on only intrinsic and social rewards. Adamic et al. have taken a close look at user behavior and content in YA [1]. They have characterized the distribution of question topics and demonstrated that responders who primarily respond to a small set of related topics have expertise in that topic and are therefore more likely to provide answers that are rated highly. The incentive mechanism used by YA is point-based; points are a non-monetary reward and they are assigned according to user behaviors such as asking a question, answering a question, providing an answer that is selected as the best answer. Jain et al. have performed a detailed analysis on the YA incentive mechanism and proven that the best answer scoring rule does not always reward responders appropriately and they suggest *approval voting* or *asker-distributed-the-points* rules [8].

The decision-theoretic framework for QA forums presented here takes a different approach for motivating responses. Like other QA systems it assumes that responders gain some intrinsic or social reward for participating, but the framework is designed to lower the responders time investment and increase the questioners confidence in the provided answers. The core of the DT framework is a recommender system. This recommender selects potential responders to answer a given question. This lowers responders time investment because they do not need to search through a list of questions to find one that they are willing and capable of answering. Moreover, this recommendation system increases questioner confidence in the responses because the recommendation algorithm is designed to identify the most appropriate experts (responders) given a question.

Expertise discovery is fundamental to recommending the best responders. There are two sources information from which to model expertise: content-based and link-based [2]. Content-based expertise modeling analyzes the word usage by responders in order to build a model of the responders expertise. This has been done in the context of clustering documents into topics by Griffiths and Steyvers [11]. Additionally, Zhang et al. have developed the QuME algorithm which identifies expertise based on matching keywords [16]. Link-based expertise identification methods rely on evaluating the link structure between questioners and responders. Zhang et al. tested a number of network based ranking algorithms on data from the Java Forum, showing that link information can be used to identify expertise nearly as well as human raters [15]. Jurczyk and Agichtein applied a variation of the HITS algorithm to the much broader Yahoo! Answers data set and were able to identify authoritative users, or expert

responders [9]. The generative model proposed here leverages both content and link information in order to best identify expertise in a QA forum.

IV. Problem Description

The Q&A recommendation problem we propose can be described as follows: given a question by a user on a Q&A forum, identify users on the forum that are most capable of providing a satisfactory answer to the question. Previous historical information about interactions on the Q&A forum is available to train the system.

More specifically, we assume that the training data is available to us in the form of individual question-answers (QAs), and for each QA, we have the following information: a) a unique questioner id, b) text of the question, c) ids of all responders, d) the text of each responder's response, and e) some information indicating which answers were found satisfactory by the questioner. Then, in the test step, we are provided with the identity of the questioner, and the text of the question. The task is to suggest suitable responders for the question. Then the actual responders are revealed, along with the text of their answers.

In the case of a live test, it is possible to directly contact the suggested responders to judge their interest in the question. However, we currently test our system using historical data derived from Q&A forums. In this case, we evaluate the quality of our recommendations by how many of our recommended responders had originally answered the question.

V. Expertise Modeling for Recommendation

Most approaches to the recommendation problem can be divided into two categories: a) content-based filtering, and b) collaborative filtering. In content-based filtering, users are modeled based on what they have been interested in, or liked, in the past. In collaborative filtering, new recommendations are made to a user based on the interests of other users identified as most similar to them. In the Q&A forum setup, we have, for each responder, content in the form of text of the questions he/she chose to answer, as well as the text used by him/her to answer these questions. We also have collaborative information about the other responders that chose to answer the same question as the responder, as well as the questioner.

A simple content-based approach to the expert recommendation problem would be to build a text-based profile, for example a term frequencyinverse document frequency (TFIDF) profile, for each responder. Then, when the system receives a new question, the question text could be compared to all the responder profiles using a similarity measure such as the cosine score, and the responders with most similar profiles could be recommended the question. This is a common approach with respect to expertise modeling and recommendation, taken by Zhang [16] and Godil [6].

However this approach, which treats the expertise identification problem as a document retrieval problem, suffers from a serious drawback: an expert is very different from a document in the sense the exact words used by a responder are heavily contingent on the questions he/she chose to answer, and do not cover all the information that a responder has, or the topic he/she may be knowledgeable about. For this reason a simple text profile based approach is not sufficient for the purpose of modeling human expertise.

To overcome this drawback, we introduce an alternate approach to expertise modeling, which models user expertise in terms of *topics*, instead of words. A topic can be seen as a higher-level concept over words, and is modeled as a distribution over words. Hence, two questions may belong to the same topic even though they may have no words in common. Similarly, an expert may be recommended a question even though there is no match in terms of profile words, if the question is judged as belonging to a topic the expert is interested in. Two words that are synonymous will have similar distributions over topics for two reasons. First, they are likely to co-occur in the same document (question and corresponding answers). Secondly, they are likely to co-occur with many of the same words.

In the next section, we introduce a generative model for a collection of question-answers in a Q&A system. Learning the parameters of this generative model enables us to identify the topics of interest to various authors, as well as topic-word distributions.

A. Generative model for a Q&A dataset

A basic outline of a generative model for any online forum where people might gather for a discussion, or to exchange information can be constructed as follows: at each timestep, a topic is generated from a distribution over topics, which might have its own prior distribution. Then, some (question) words are generated related to the topic. The topic distribution may or may not be independent of the original author of the post, depending on how closely people stick to their topic of interest. Following this, a set of responders are chosen from a distribution, based on the topic, and each of these responders generate further words. The words generated by the responders are related to the topic, but may or may not be seen as drawn from the same distribution as the topic. For example, if users have strong personal opinions, or try to draw the discussion in some favoured direction, this might need to be modeled as each user having its own word distribution for each topic, or the words as drawn from a mixture distribution of the original topic distribution, and a word distribution related to the user.

The model outlined above will be expensive to model due to the large number of parameters involved. We simplify the model considerably, reducing it to a finite mixture model in the process. These simplifications reduce the number of parameters considerably, while, we believe, still providing important insight into the dataset. We describe our generative model below:

We assume that the number of unique words p, the number of topics $|T|$, and the number of unique responders s is known in advance. Let the words be labeled $1, \ldots, p$ and the users $1, \ldots, s$, arbitrarily. At each timestep, a topic t is generated from a mutinomial distribution τ over topics,

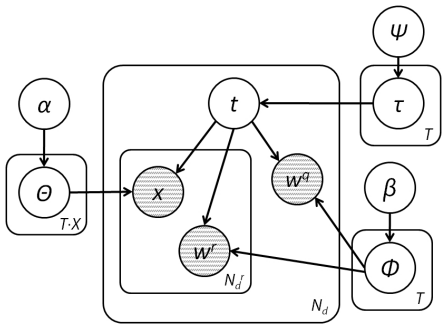

Fig. 1. Generative Model for Question Answer Recommender

and a vector $\vec{w^q} = \{w_1, \ldots, w_p\}$ is generated, where w_i, is the count of word labeled i in the generated words. The words are generated from ϕ_t, a multinomial distribution over words corresponding to topic t. Following this a responder vector $\vec{x} = \{x_1, \ldots, x_s\}$ is generated from θ_t, a multinomial distribution over users for topic t, where x_i is the number of time the user labeled i responded. Each of the users in x in turn generates words based on the topic t. Here, we make the important simplifying assumption that the words generated by a responder as part of the answer are drawn from the same distribution ϕ_t as the topic.

This assumption can be understood as saying that the words used in the answer to a question by a responder depend only on the topic of the question, and do not depend on any attributes of the responder. We believe this to be a reasonable assumption in Q&A forums where factual information is exchanged for the most part, or even in forums where personal opinions are expressed but the vocabulary used does not differ very much from user to user. It may not hold true in forums such as blogs or discussion forums, where responses to topics are much longer and more personal, and people may have favorite topics they might try to steer the topic toward. However, we believe that that level of model complexity is not required for Q&A forums.

Figure 1 displays the generative model described above in plate notation. The shaded variables are the observed variables, while the unshaded variables are the hidden variables. Also, α, β and γ are symmetric Dirichlet priors, used for smoothing. We set $\alpha = \beta = \gamma = 1$. We also make the simplifying assumption that the total number of words and users generated for each question is independent of τ, θ and ϕ, and hence their randomness can be ignored in our discussion. Also, since we assume that the words generated by the responders depend solely on the topic, we can write the words generated by all responders as a vector $\vec{w^r} = \{w_1, \ldots, w_p\}$. We write $\vec{w} = \vec{w^q} + \vec{w^r}$.

B. Learning parameters of the model

It is easy to see from figure 1 that our generative model is essentially a mixture model with a finite number of components, where the number of components is the number of topics we expect to see in the dataset. Let the number of such components/topics be g. Let the total number of unique Q&A interactions in the dataset D be n, where the j^{th} such interaction is referred to as d_j. Then, we assume, there is associated with each d_j a hidden vector z_j of length g, where $z_{ij} = 1$ if d_j is about topic i. We write $\vec{\theta} = \{\theta_1, \ldots, \theta_g\}$, $\vec{\phi} = \{\phi_1, \ldots, \phi_g\}$, and $\epsilon = (\bar{\theta}, \bar{\phi})$. Then, assuming we know z_j for all d_j, we can write the log likelihood of ϵ given D as:

$$\log_D L(\epsilon) = \sum_{i=1}^{g} \sum_{j=1}^{n} z_{ij} \{\log \tau_i + \log P(\vec{w_j}|\theta_i) + \log (P(\vec{x_j}|\phi_i)\}$$

We use the expectation maximization (EM) algorithm [5] to estimate the hidden variables z_i, and the parameters τ and ϵ. The derivation of the EM algorithm for the model is fairly straightforward. We give the E-step and the M-step below:

E-Step:

Given a guess for τ and ϵ, the expected value of z_{ij} is given by:

$$z_{ij} = \frac{\tau_i \cdot P(\vec{w_j}, \vec{x_j}|\epsilon_i)}{\sum_{h=1}^{g} \tau_h \cdot P(\vec{w_j}, \vec{x_j}|\epsilon_h)}$$

M-step:

Given expected values of z_j, we can estimate τ, θ and ϕ as follows:

$$\tau_i = \sum_{j=1}^{n} \frac{\psi + z_{ij}}{\psi|T| + n}$$

$$\theta_{ik} = \frac{\alpha + \sum_{j=1}^{n} z_{ij} x_{jk}}{\alpha s + \sum_{j=1}^{n} \sum_{k'=1}^{s} z_{ij} x_{jk'}}$$

$$\phi_{ik} = \frac{\beta + \sum_{j=1}^{n} z_{ij} w_{jk}}{\beta p + \sum_{j=1}^{n} \sum_{k'=1}^{p} z_{ij} w_{jk'}}$$

VI. Evaluation Metrics

Any evaluation of Q&A recommender systems needs to take into account the experience of users from both perspectives: as questioners and as responders. As questioners, they would like highly satisfactory responses. As responders, they would like not to be overloaded with too many questions, or be recommended questions they are not interested in. In particular, if they are high quality experts, there is a risk that they might be recommended too many questions in a bid to provide satisfactory responses to questioners, which might result in reduced participation from them.

We introduce two new metrics, *responder load* and *questioner satisfaction*, to measure the quality of Q&A recommenders from both of these perspectives. Both of these metrics are variations of *precision* and *recall*, metrics commonly used in information retrieval [13] and recommender system research [12].

It is possible to define precision and recall for a user in a Q&A system from two perspectives: as a questioner and as a responder. So we can define four metrics of interest in total. However, the action governing the quality of all four metrics is the same: each time a new question is introduced in the system, the recommender makes a decision to contact

a subset of responders in the system and recommend the question to them. The decision to answer the question is made by each individual responder and cannot be controlled by the recommender. So the decision made by the recommender to recommend a question has to take into account both the possible impact on questioner metrics as well as responder metrics.

A. Responder Precision and Recall

Let all the participants in the QA forum be represented by X. Any $x \in X$ can be a questioner or a responder/answerer. Let the user x's responder precision be written as π_x^a, and responder recall as ρ_x^a. Then,

$$\pi_x^a = \frac{R_x^+}{R_x^+ + N_x^+}$$

$$\rho_x^a = \frac{R_x^+}{R_x^+ + R_x^-}$$

Here, the right-hand-side terms are as defined in Table I. Table I can be understood as follows: R_x implies that the responder x liked the question, i.e., responded to the question. N_x means that the responded x did not like the question. A superscript of $+$ means the recommender system suggested the question to the responder x. A superscript of $-$ suggests that it did not suggest the question to the responder. For example, R_x^+ is a count of the number of questions that are both liked by responder x and recommended to x.

For a given responder, responder precision is the ratio of the number of questions that were recommended to a responder, that the responder answered, to the total number of questions recommended to the responder. We believe this to be an important measure of the quality of the recommender, as a recommender that recommends too many irrelevant questions will drive away responders.

The recall for a responder measures how many of the questions the responder answered were recommended by the system. It is a measure of how well the recommender covers all the interests of the questioner, and while still important, we consider it relatively less significant.

B. Questioner Precision and Recall

Similarly, let the user x's questioner precision be written as π_x^q, and questioner recall as ρ_x^q. Then,

$$\pi_x^q = \frac{U_x^+}{U_x^+ + I_x^+}$$

$$\rho_x^q = \frac{U_x^+}{U_x^+ + U_x^-}$$

Here the right-hand-terms are as defined in Table II.

Questioner precision measures how many of the responders recommended by the system provided satisfactory answers. This is also relatively unimportant: a user will not usually mind extra answers so long as he/she is receiving a sufficient number of answers that are satisfactory. There may be problems in extreme cases, such as when a particular user is spammed, but this problem might be handled in other ways, such as allowing questioners to ban specific responders from their questions, or rank answers based on responder quality, or responder history.

Questioner recall measures how many of the answers/responders of interest to the questioner, the recommender was able to identify in advance. This metric is more important than questioner precision, as it indicates the degree to which questioners can depend on the recommender to find the most suited responders for a question.

We focus on Responder Precision and Questioner Recall when testing our recommender system. In the next section, we describe a small variation of Responder Precision, and rename Questioner Recall. These re-descriptions, we believe, make these metrics more intuitive to understand in terms of recommender system behavior.

C. Responder Load and Questioner Satisfaction

We define *Responder Load* λ_x for user x as $\lambda_x = 1 - \pi_x^a$, or:

$$\lambda_x = \frac{N_x^+}{R_x^+ + N_x^+}$$

The higher the value of λ_x, the greater the number of questions a responder has to read through to find questions of interest to him/her. In that sense, it is a measure of the load on the responder.

We define *Questioner Satisfaction* as $\sigma_x = \rho_x^q$, or:

$$\sigma_x = \frac{U_x^+}{U_x^+ + U_x^-}$$

σ_x measures what fraction of the answers founds satisfactory by a questioner were from responders contacted by the recommender. It can be seen as a measure of how satisfied questioners will be with the Q&A system if the responders relied entirely on the recommender to provide them with interesting questions.

The overall quality of a QA recommender could be measured as $\sum_{x \in X} w_x \lambda_x$, and $\sum_{x \in X} w_x \sigma_x$. w_x could be set based on some criteria, or set as $\frac{1}{|X|}$, to get the average values. In the experiments presented here, There is a tradeoff between λ and σ. For example, by recommending all responders in the system, we can move the satisfaction σ_x to 1 for all questioners. However, this will have an adverse impact on the questioner load, as most of the questions we suggest to a responder, he/she will not find interesting. Hence, the basic challenge in the problem is to provide quality answers to questioners, without overloading the better experts among responders, and to manage this tradeoff in a reasonable way.

VII. Expected Utility

We create a composite utility metric to manage the tradeoff between λ and σ in a principled manner.

$$U = \sum_{x \in X} w_x((\bar{\lambda} R_x^+ - N_x^+) + (\bar{\sigma} U_x^+ - U_x^-)) \qquad (1)$$

The utility function expresses the tradeoffs we believe are acceptable for the system we build. $\bar{\lambda}$ and $\bar{\sigma}$ can be seen as

	Liked question / Answered question	Did not Like / Did not Answer
Recommended	R_x^+	N_x^+
Not Recommended	R_x^-	N_x^-

TABLE I
METRICS TABLE FOR RESPONDER x

	Liked Answer / Upvoted Answer	Did not like Answer / Ignored Answer
Recommended	U_x^+	I_x^+
Not Recommended	U_x^-	I_x^-

TABLE II
METRICS TABLE FOR QUESTIONER x

setting the tradeoff we find acceptable between questioners and responders. For example, setting $\bar{\lambda}$ to 10 would suggest that the system is willing to tolerate 10 incorrect recommendations for a single correct recommendation. For example, in a system where experts are paid, high values of $\bar{\lambda}$ and $\bar{\sigma}$ might be acceptable. On the other hand, in a voluntary system where experts are generally busy, lower values of $\bar{\lambda}$ and $\bar{\sigma}$ might be a good idea. We use two parameters $\bar{\lambda}$ and $\bar{\sigma}$ instead of one, so as to effectively manage the tradeoff between questioners and responders. This is discussed further in the next section.

VIII. UTILITY BASED RECOMMENDATION

Definitions:
Availability: For a given agent/responder x, we define its availability in a topic t, v_x^t, as the probability that the agent/responder will answer any given question in topic t.
Expertise: For a given agent/responder x, we define its expertise in a topic t, e_x^t, as the probability that the agent's/responder's answer will be rated as satisfactory, given that the responder answers the question. We make the simplifying assumption that the rating provided by the questioner as satisfactory/unsatisfactory does not depend on the questioner, and depends only on the quality of the answer.

Thus, for a given question from topic t, the probability that a user x will answer the question is v_x^t, and the probability that he/she will answer the question satisfactorily as $v_x^t \cdot e_x^t$.

Let there be a question asked by a questioner q. Then, suppose our recommender contacts/recommends responder a. If a answers the question, R_a^+ increases by 1. Hence U_a increases by $\bar{\lambda}$. If a does not answer the question, N_a^+ increases by 1, and the overall utility of a, U_a, decreases by 1. Then the expected change in utility U_a for responder a, if he/she is recommended a question by user q, written as $\Delta E(U_a^{(q,a)})$, is given by:

$$\Delta E(U_a^{(q,a)}) = \bar{\lambda} v_a - (1 - v_a)$$
$$= (\bar{\lambda} + 1) v_a - 1$$

Similarly, the expected change in utility for the questioner, $\Delta E(U_q^{(q,a)})$, can be calculated:

$$\Delta E(U_q^{(q,a)}) = \bar{\sigma} v_a e_a - (1 - v_a e_a)$$
$$= (\bar{\sigma} + 1) v_a e_a - 1$$

The overall expected change in utility can then be calculated as:

$$\Delta E(U^{(q,a)}) = w_a \cdot \Delta E(U_a^{(q,a)}) + w_q \cdot \Delta E(U_q^{(q,a)}) \quad (2)$$

Expanding this gives:

$$\Delta E(U^{(q,a)}) = w_a \cdot ((\bar{\lambda}+1) v_a - 1) + w_q \cdot ((\bar{\sigma}+1) v_a e_a - 1) \quad (3)$$

Our recommendation approach can then be summed up as follows:

1) Given a question by questioner q: for each candidate responder a, estimate v_a and e_a.
2) Calculate $\Delta E(U^{(q,a)})$ based on these estimates of v_a and e_a.
3) Recommend the question to all responders for whom $\Delta E(U^{(q,a)})$ is positive.

IX. EXPERIMENTS

We have performed experiments to demonstrate a working implementation of our recommender based QA framework. 5000 questions and their answers from each of three different subject categories in Yahoo! Answers have been crawled, parsed, processed. These categories are Astronomy and Space, Books and Authors, and Wrestling. The collected data represents approximately one month of activity within each category. These categories were chosen to represent a broad spectrum of the type of content available on YA. After the pages were stripped of html, we applied stemming and stopping algorithms to remove suffixes and unimportant words.

The first experiment compares the effectiveness of three different methods of identifying and modeling expertise. The first method uses a basic information retrieval (IR) algorithm that calculates the cosine similarity between the words used by an author in historical QA data and the words contained in a question. This is included as a baseline measurement. The second method uses K-Means clustering [7] to group question-answer documents into clusters. Users are then assigned a probability weight in each cluster based on the fraction of questions answered in the cluster. Given a new question, users are recommended based on the normalized weighted sum of the similarity of the question to each cluster centroid,

multiplied by the probability of response (availability) of each user in the cluster.

The third method uses our generative model to discover the author-topic distributions. Given a new question, a probabilistic estimate is made of the topic of the question, and responders are recommended based on their marginalized probability of responding across all topics, using θ.

Our expertise models are built on a training set of 4000 questions and answers. The remaining 1000 questions and answers form the test set. For each of these questions in the test set we apply the decision-theory framework to calculate an estimated change in utility for each potential responder. The values for w_a and w_q represent the historic prolificacy of the responder, or the normalized number of times that responder and answerer responded in the past. We then recommend the responders with a positive estimated change in utility. The values for $\bar{\lambda}$ and $\bar{\sigma}$ are chosen to tune the recommender performance. These values are chosen to encourage a large number of recommendations and therefore high questioner satisfaction at the expense of responder load. The recommender performance is analyzed using our metrics of responder load and questioner satisfaction. We introduce one more simple metric called *weak satisfaction* that indicates whether the recommender was able to select at least one of authors who actually responded. $\bar{\lambda}$ and $\bar{\sigma}$ have been selected to fix *responder load* near 0.95. With a fix load, it is easier to compare the tradeoffs made when evaluating questioner satisfaction. We make a simplifying assumption that all actual responses are satisfactory. Yahoo! Answers collects best-answer information that indicates the single most satisfactory answer. However, the unpredictability of the dataset makes recommending the single most satisfactory response difficult, and also, marking a single response as satisfactory does not imply that all other responses were unsatisfactory. Table III displays the experimental parameters and results of our experiment to examine the impact of expertise modeling on recommendation.

The second experiment compares the performance of a recommender based on a decision-theoretic framework versus a very simple recommender that makes recommendations based only on the strength of an expertise match, neglecting the tradeoff between load and satisfaction. Table IV contains the results of this comparison. The *Best Match* algorithm simply recommends the top 25 responders according to the expertise information from the *EM* algorithm. This is done to simulate a naive recommendation system that does not use a decision-theoretic framework.

A. Analysis

Yahoo! Answers is a noisy and unpredictable dataset. Over a set of 5000 questions representing one month of activity in one category, 11588 unique users participated. Of these users, 4890 responded to a question only once, and 4723 never responded to a single question, but only asked questions. In addition, many users leave the system after a short period of time. 2319 new users appeared in our test data set of 1000 questions.

Running an offline test of a recommender is very difficult. Our results show that our system was able to correctly recommend at least one responder more than half the time (weak questioner satisfaction) while maintaining a load of $\lambda < .95$. This means that of every 20 recommendations one user responded to the question. Given the bulletin board structure of YA, it is certain that responders rarely see every available question. In a live recommender test, we predict a much higher percentage of responses from recommended responders because we can assume the responder sees the question and knows it has been recommended based on his/her expertise. Also, with a live test it would be possible to measure the questioner's satisfaction with any given response, leading to a more accurate measure of the recommender performance regarding questioner satisfaction.

Table III contains the results of the first experiment. A *responder load* of 0.95 indicates that of 20 questions recommended to a user, he/she responded to one. This number is artificially high because we have an offline test. The results show that the *clustering* and *EM* algorithms outperform the basic information retrieval algorithm according to questioner satisfaction in nearly every case. This evidence supports using more sophisticated techniques for discovering responder expertise. The *EM* and *clustering* algorithms performed very similarly across all three data sets. While recommendation was more successful with some data sets, the relative performance of the algorithms was preserved.

Table IV compares two types of recommender system. First is the decision-theory based method described in Section VIII, and the second is a simple *Best Match* algorithm. While this is just a single example, it shows that a more sophisticated utility-based recommender can outperform a simple recommender, even when they are supplied the same expertise information. Our experiments show that the decision-theoretic framework only slightly outperformed the simple best-match recommendation algorithm. We hypothesize that interactive experiments run on a live dataset will better demonstrate the load balancing features of the decision-theoretic framework.

X. Conclusions

We have developed a recommender for selecting the most appropriate responders given a question. This recommender is the core of a question and answer forum under development that is designed to encourage expert participation. The two primary contributions of this work are a finite mixture model based approach for characterizing the production of content in an online question and answer forum and, a decision theoretic framework for recommending expert participants while maintaining questioner satisfaction and distributing responder load. Our generative model uses word content information and collaborative information to build models of users expertise, which are employed during recommendation. We have also developed two new metrics: responder load and questioner satisfaction. These metrics are used to evaluate the performance of our recommender system on datasets harvested

Expertise Modeling Parameters		Recommender Performance		
Astronomy & Space		**Responder Load**	**Questioner Satisfaction**	**Weak Satisfaction**
Info Retrieval	$\bar{\lambda} = 35$ $\bar{\sigma} = 20$	0.9485	0.1187	0.3570
Clustering	$\bar{\lambda} = 50$ $\bar{\sigma} = 20$ $K = 30$	0.9480	0.2682	0.6440
EM	$\bar{\lambda} = 250$ $\bar{\sigma} = 250$	0.9559	0.2390	0.6567
Books & Authors		**Responder Load**	**Questioner Satisfaction**	**Weak Satisfaction**
Info Retrieval	$\bar{\lambda} = 15$ $\bar{\sigma} = 5$	0.9743	0.0186	0.1141
Clustering	$\bar{\lambda} = 35$ $\bar{\sigma} = 20$ $K = 15$	0.9700	0.0079	0.0531
EM	$\bar{\lambda} = 200$ $\bar{\sigma} = 300$	0.9701	0.0419	0.1722
Wrestling		**Responder Load**	**Questioner Satisfaction**	**Weak Satisfaction**
Info Retrieval	$\bar{\lambda} = 25$ $\bar{\sigma} = 10$	0.9740	0.0472	0.2472
Clustering	$\bar{\lambda} = 75$ $\bar{\sigma} = 100$ $K = 30$	0.9748	0.2115	0.7050
EM	$\bar{\lambda} = 300$ $\bar{\sigma} = 200$	0.9752	0.0922	0.4424

TABLE III
EXPERTISE MODELING COMPARISON

Astronomy & Space		**Responder Load**	**Questioner Satisfaction**	**Weak Satisfaction**
DT Framework (EM)	$\bar{\lambda} = 250$ $\bar{\sigma} = 250$	0.9559	0.2390	0.6567
Best Match	Top 25	0.9467	0.2102	0.6046

TABLE IV
RECOMMENDATION ALGORITHM COMPARISON

from Yahoo! Answers. Three methods of constructing expertise models are compared: a simple information retrieval approach, clustering words to discover their distribution over topics, and expectation maximization based on our generative model. In addition our decision-theoretic based recommender is compared to a simple best-match configuration. Experiments across several topic domains demonstrate our systems ability to predict responder identities and suggest new responders.

In the future, we intend to build a full QA system and test our recommender using a live, instead of static, dataset. A substantial component of this planned system is an incentive mechanism for encouraging expert participation. Such an incentive mechanism would use responder and questioner feedback along with link analysis to guide recommendation. The ideas presented here are the first steps in designing a question and answer forum that is capable of identifying and incentivizing its most valuable contributors while routing questions to the most appropriate responder(s). An easy question does not require an expert, and from a social welfare perspective it would be harmful to recommend one. An ideal QA forum would maximize user satisfaction both as a questioner and a responder.

Acknowledgements This research is sponsored in part by the Office of Naval Research Project #N00014-05-1-0857-UTARQ-M. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon.

REFERENCES

[1] L. A. Adamic, J. Zhang, E. Bakshy, and M. S. Ackerman, "Knowledge sharing and yahoo answers: everyone knows something," in *WWW '08: Proceeding of the 17th international conference on World Wide Web*. New York, NY, USA: ACM, 2008, pp. 665–674.

[2] G. Adomavicius and A. Tuzhilin, "Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions," *Knowledge and Data Engineering, IEEE Transactions on*, vol. 17, no. 6, pp. 734–749, June 2005.

[3] Y. Benkler, "Coase's Penguin, or, Linux and the Nature of the Firm." *Yale Law Journal*, vol. 112, no. 3, pp. 367–445, 2002.

[4] J. Bennett and S. Lanning, "The netflix prize," in *Proceedings of KDD Cup and Workshop*, vol. 2007, 2007.

[5] A. Dempster, N. Laird, and D. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 1–38, 1977.

[6] H. Godil, *Finding Experts by Modeling Domain Expertise*. University of Texas at Austin, 2006.

[7] J. Hartigan, *Clustering algorithms*. John Wiley & Sons, Inc. New York, NY, USA, 1975.

[8] S. Jain and D. Parkes, "Designing Incentives for Online Question and Answer Forums," in *to appear in Proceedings of the 10th ACM conference on Electronic commerce*, 2009.

[9] P. Jurczyk and E. Agichtein, "Discovering authorities in question answer communities by using link analysis," in *CIKM '07: Proceedings of the sixteenth ACM conference on Conference on information and knowledge manageme nt*. New York, NY, USA: ACM, 2007, pp. 919–922.

[10] J. Pouwelse, P. Garbacki, D. Epema, and H. Sips, "Pirates and Samaritans: A decade of measurements on peer production and their implications for net neutrality and copyright," *Telecommunications Policy*, vol. 32, no. 11, pp. 701–712, 2008.

[11] M. Rosen-Zvi, T. Griffiths, M. Steyvers, and P. Smyth, "The author-topic model for authors and documents," in *AUAI '04: Proceedings of the 20th conference on Uncertainty in artificial intelligence*. Arlington, Virginia, United States: AUAI Press, 2004, pp. 487–494.

[12] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Analysis of recommendation algorithms for e-commerce," in *EC '00: Proceedings of the 2nd ACM conference on Electronic commerce*. New York, NY, USA: ACM, 2000, pp. 158–167.

[13] A. Singhal, "Modern information retrieval: A brief overview," *IEEE Data Engineering Bulletin*, vol. 24, no. 4, pp. 35–43, 2001.

[14] L. von Ahn and L. Dabbish, "Designing games with a purpose," *Commun. ACM*, vol. 51, no. 8, pp. 58–67, 2008.

[15] J. Zhang, M. S. Ackerman, and L. Adamic, "Expertise networks in online communities: structure and algorithms," in *WWW '07: Proceedings of the 16th international conference on World Wide Web*. New York, NY, USA: ACM, 2007, pp. 221–230.

[16] J. Zhang, M. S. Ackerman, L. Adamic, and K. K. Nam, "Qume: a mechanism to support expertise finding in online help-seeking communities," in *UIST '07: Proceedings of the 20th annual ACM symposium on User interface software and technology*. New York, NY, USA: ACM, 2007, pp. 111–114.