FINDING EXPERT USERS IN COMMUNITY QUESTION ANSWERING SERVICES USING TOPIC MODELS

by

Fatemeh Riahi

Submitted in partial fulfillment of the
requirements for the degree of
Master of Computer Science

at

Dalhousie University
Halifax, Nova Scotia
February 2012

DALHOUSIE UNIVERSITY

FACULTY OF COMPUTER SCIENCE

The undersigned hereby certify that they have read and recommend to the Faculty of Graduate Studies for acceptance a thesis entitled "FINDING EXPERT USERS IN COMMUNITY QUESTION ANSWERING SERVICES USING TOPIC MODELS" by Fatemeh Riahi in partial fulfillment of the requirements for the degree of Master of Computer Science.

Dated: February 29, 2012

Supervisor: _____

Readers: _____

_____

DALHOUSIE UNIVERSITY

DATE: February 29, 2012

AUTHOR: Fatemeh Riahi

TITLE: FINDING EXPERT USERS IN COMMUNITY QUESTION ANSWERING SERVICES USING TOPIC MODELS

DEPARTMENT OR SCHOOL: Faculty of Computer Science

DEGREE: M.C.Sc. CONVOCATION: May YEAR: 2012

Signature of Author

Table of Contents

List of Tables

List of Figures

Abstract

Community Question Answering (CQA) websites provide a rapidly growing source of information in many areas. In most CQA implementations there is little effort in directing new questions to the right group of experts. This means that experts are not provided with questions matching their expertise. In this thesis, we propose a framework for automatically routing a newly posted question to the best suited expert. The purpose of this framework is to decrease the waiting time for a personal response.

We also investigate the suitability of two statistical topic models for solving this issue and compare these methods against more traditional Information Retrieval approaches. We show that for a dataset constructed from the Stackoverflow website, these topic models outperform other methods in retrieving a set of best experts. We also show that the Segmented Topic Model gives consistently better performance compared to the Latent Dirichlet Allocation Model.

List of Abbreviations and Symbols Used

EM	Expectation Maximization
LDA	Latent Dirichlet Allocation
LM	Language Model
LSI	Latent Semantic Indexing
NLP	Natural Language Processing
PLSI	Probabilistic Latent Semantic Indexing
STM	Segmented Topic Model
TF-IDF	Term Frequency Inverse Document Frequency
UPM	User Persona Model

Acknowledgements

I am heartily thankful to my supervisors, Dr. Milios and Dr. Shafiei, their encouragement, guidance and support from the initial to the final level enabled me to develop an understanding of the subject.

My deepest gratitude goes to my family for their unflagging love and support throughout my life; this dissertation would have been simply impossible without them. I am indebted to my father for his care and love. I cannot ask for more from my mother, as she is simply perfect. I have no suitable word that can fully describe her everlasting love to me.

Lastly, I offer my regards and blessings to all of those who supported me in any respect during the completion of the project.

Chapter 1

Introduction

Community Question-Answering (CQA) services contain millions of questions and answers and provide a valuable resource that cannot be easily obtained using web search engines. Providing good quality answers to users' questions through collaboration of a community of experts is the main purpose of these services. Voting, badges and reputation are examples of mechanisms provided by some CQA services to assure the quality of questions and answers.

In current CQA services, a user who has an information need is required to either (i) wait for other users to post answers to the question which may take several days and sometimes results in incorrect or spam answers (ii) or use the archives of CQA sites. These archives often contain restricted answer sets and the user has to deal with the word-match constraint between her formulated question and archived questions [25].

The main problem of CQA services is the low participation rate of the users. It means that only a small portion of users are responsible for answering a notable number of questions. Two main reasons of low participation are: (i) most users are not willing to answer questions or are not experts, (ii) those users willing to answer questions are not aware of the new questions of interest to them [14].

Developing a system capable of finding experts for newly posted questions can contribute to the creation of high-quality answers for questions and mitigate the problem of low participation rates. The goal of expert finding is to return a ranked list of experts with special knowledge on a given topic. An important part of an expert-finding task is how to model the expertise and the interest of a user based on her answering history.

Common methods for finding experts can be divided into two categories. The first category searches for relevant answers for a given question and then retrieves a ranked list of users based on their contribution to those answers (Fig. 1.1). The second

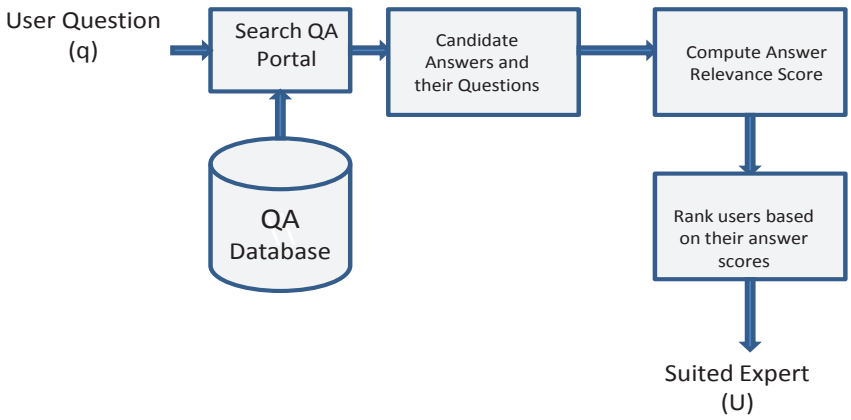


Figure 1.1: First category of finding experts. Using the ranked list of documents to locate the best expert

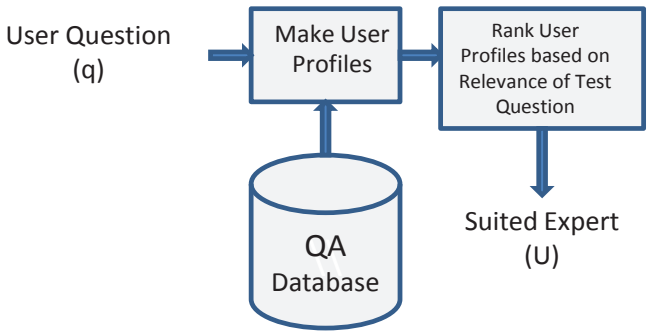


Figure 1.2: Second category of finding experts. Using profiles to locate the best expert

category builds a profile for each expert based on her activity and past answers and then uses these profiles to find experts. Our research falls into the second category by building a profile for each user and finding experts using these profiles (Fig. 1.2).

Most of the current works in the latter category model user profiles by using classical information retrieval approaches. These approaches use lexical similarity measures and retrieve good results if enough word overlap exists. However, there is not often much word overlap between new questions and user profiles, therefore these approaches may not lead to satisfactory results.

In this thesis, we focus on the second cause of low participation rate in CQA mentioned above. Our objective is to route new questions to the best suited experts. We model the interests of users by following their answering history in the CQA services. For

each user, a profile is created by combining those questions answered by the user for which she has been selected the best answerer. Based on the user profiles, the relation between the answerer and a new question is measured by using a number of different methods. These methods include language models with Dirichlet smoothing, TF-IDF, the Latent Dirichlet Allocation (LDA) and Segmented Topic Model (STM).

Questions posted on CQAs are usually short in length. A question may be semantically similar to a user profile but still lexically very different. Therefore, an expert recommender system that is capable of capturing the semantic similarities between the question and user profiles may achieve better results.

LDA and STM model the latent topics in user profiles to capture the semantic structure of user profiles. In LDA, all questions answered by a user are concatenated together and build the user profile. However, a user is likely to have answered questions in different topics. STM, on the other hand, is more focused on taking advantage of the structure of CQAs and extracts more semantic information from the profiles. STM treats each question individually while considering all questions answered by the user as her profile. Our experimental results indicate that STM performs much better than LDA in retrieving a candidate set of best experts for a question.

Evaluating systems for expert finding is not a simple task. In our dataset, which is a snapshot of Stackoverflow, we have the actual best answerer for each test question and we use it to evaluate performance of a method. However, it is quite likely that other users returned by the system are also good answerers for the questions. In order to detect the relevancy of other returned users to the test question, a user study would be required.

Chapter 2

Background

In this chapter, we review some of the research on community question answering, expert recommendation and topic analysis using statistical models.

2.1 Community Question Answering

In the past few years, Community Question Answering websites such as Yahoo! answers have been building large archives of questions and answers [1, 14, 29, 21].

Research on community question answering has seen a significant growth. One of the main goals of this research is to decrease the lag time for a response. Relying on the available archive, one can approach this problem by either finding similar questions or relevant answers.

When relying only on the questions available in the archive, the objective is to find similar questions previously answered by the QA community.
CQAs usually provide question search services that search the archive for the previously asked questions. The first row of Table 2.1 shows a sample query question, and the second row shows the result of question search on Stackoverflow. Many methods have been introduced for tackling this problem. A retrieval model based on translation probabilities learned from the archive of questions and answers is able to find semantically similar questions with relatively little word overlap [17]. An automatic method for finding questions that have the same meaning calculates question-question similarities by using the corresponding answers and questions [16]. They also developed two similarity measures based on language model and compared them with the traditional similarity measures.

To complement question search, a novel application called question recommendation is introduced in [8]. They consider a question as a combination of question topic and question focus. Question topic characterizes users' interest (e.g., Perl) and question focus represents certain aspects of a user interest (e.g. reading file). The goal of

Table 2.1: Question Search vs. Question Recommendation

information need:
how to read a file in perl
Question Search:
Read and write a text file in perl
Question Recommendation:
perl read a file into an array
How can I read from continuously updated file in Perl?
Read binary file in perl

question recommendation is to make users familiar with other existing aspects similar to the question topics that might be interesting for them. They tackled this problem by representing questions as graphs of topic terms and then ranking recommendations on the basis of these graphs.

Including the answers available in the archive, the main purpose is to find a right answer in QA archive for a given question. To build an answer finding system, four statistical techniques are used in [4] including TF-IDF, adaptive TF-IDF, query expansion and statistical translation. A semantic knowledge base (WordNet) is used in [7] to improve the ability of classical information retrieval approaches in matching questions and answers. Additionally, non textual features are used to improve the answer search quality in [18].

2.2 Expert and Question Answerer Recommendation

Compared to the previous problem of retrieving relevant questions and answers for a new question, there are fewer works aiming to solve the problem of finding the best answerers or experts for a new question. The task of expert recommendation is predicting the best users who can answer a newly posted question. A ranked list of best answerers can be returned based on the similarity between the new question and users' history. However, there are some aspects that make expert finding different from the answer finding task. According to [1], there are cases in CQAs where users ask questions for neither expertise nor support, but only to get an opinion or to start a conversation. In such cases, expert users are not the only possible answerers. Stackoverflow is not a general forum and most of the time askers are focused on specific topics. Thus, predicting an answerer for a new question on this domain is

more similar to expert finding than merely finding a potential answerer.

To locate users with desired expertise, quantitative measures of expertise are defined in [24]. They described how to obtain these measures from a software project's change management system. They also presented evidence to validate this quantification as a measure of expertise. Two general strategies for expert searching given a document collection are presented in [3] by using generative probabilistic models. Experts are found by mining expertise from email communications in [10]. Profile-based models for expert finding on general documents are proposed in [12].

There is also some research in question answerer recommendation. A new topic model which can simultaneously discover topic distribution for words, categories and users in a QA community is introduced to find a ranked list of answer providers [14]. Latent Dirichlet Allocation model is used in [20] and it has been combined with user activity and authority information to find the best answerers.

2.3 Topic Analysis Using Statistical Models

The use of topic models for information retrieval tasks is described in [26]. They found that the combination of Dirichlet smoothed language models and topic models lead to significant improvements in retrieval performance compared to using only the language models.

The most popular model for text retrieval is the Vector Space Model [2]. However, this model suffers from high dimensionality when representing documents using the "bag of words" assumption. Latent Semantic Indexing (LSI) [9] is one way to reduce the space dimension but it lacks semantic interpretation. To overcome this problem, pLSI [15] introduces latent topics to represent documents and model the data generation process as a Bayesian network. A novel model, Author-Persona-Topic (APT), is introduced in [23] to recommend the best reviewers for a given paper by dividing authors' papers into several "personas". Each persona clusters papers with similar topical combinations. A new topic model, the author-topic model (ATM), is proposed in [30], for exploring the relationships between authors, documents, topics and words.

2.4 Overview of the Baselines

In this section, we will provide a brief description of the methods used as baseline in this thesis. These studies mostly include some of the well known algorithms used for modelling and representing text data.

2.4.1 Word-based Models

Traditional text mining applications usually use these kinds of methods and represent text by the Vector Space Model. However, this group of methods usually suffer from some problems such as high dimensionality. In the following, we briefly review two well known models of this family and discuss some of their limitations. Later on, we review some of the solutions proposed to address these limitations.

TF-IDF

The main challenge of word-based methods is data representation. A good representation approach for data may lead to the discovery of a better structure for text. The first step towards choosing a data representation is defining the meaningful unit of text: words or character N-grams.

In the vector space model, a document is shown by a vector of weights of features such as words or terms or character N-grams extracted from documents.

If the total number of documents in the corpus is d and the total number of features is t, then the corpus is represented by a matrix with dimensionality of $d \times t$. The values in this representation shows the importance of a term in documents and the entire corpus. The number of words in the vocabulary determines the length of the vector used for representing documents. Each element of this vector represents the weight of the corresponding word in the document most often defined as the frequency of that term in the document.

The most popular method that uses this weighting scheme is called "term frequency-inverse document frequency". TF-IDF is a standard measure to compute importance and relevance of a word in a document. This measure is based on the frequency of a word in the document and the inverse proportion of documents containing that word

over the entire document corpus. Words that appear only in a small group of documents will have higher tf-idf scores than other words. Basically, TF-IDF is defined as follows: given a document collection Q, a word w, and a document $q \in Q$:

$$tfidf = f_{\mathrm{w,q}} * \log(\frac{|Q|}{f_{\mathrm{w,Q}}}) \tag{2.1}$$

where $f_{\mathrm{w,q}}$ is the number of times w appears in q, $|Q|$ is the size of the corpus and $f_{\mathrm{w,Q}}$ is the total number of documents that have the word w [31].
The end result is a term-by-document matrix for the entire corpus, where columns represent terms, rows represent documents and each value in the matrix represents the tf-idf weight for the corresponding term and document. Thus, the TF-IDF reduces documents of different lengths to vectors with a fixed length.

Language Model

The main intuition of language models in natural language is that some tokens are more general than others. In this intuition, Language Models are related to traditional TF-IDF models. Similar to TF-IDF, rare terms in the corpus which occur in only a group of documents in the corpus, have a great influence on the ranking. Research in information retrieval shows that the language model approach is more effective than TF-IDF [28]. Language models represent text by using a probability distribution over strings; where strings are chosen from a fixed set of tokens. The tokens can be the words in the vocabulary. This model assumes that each word is independent of all other words in the vocabulary. The joint distribution for the entire sequence w_1, w_2, ..., w_{n} can be shown as:

$$P(w_1, w_2, ..., w_{\mathrm{n}}) = \prod_{i=1}^{n} P(w_{\mathrm{i}}) \tag{2.2}$$

In this model, a multinomial probability distribution over words in the vocabulary is used to represent a document. A new document is represented by a set of words q= $\{w_1,w_2,...,w_{\mathrm{N}}\}$ where w_{i} is a non-stop word and each question is assumed to be generated independently. Therefore, the probability of a new document being more related to a candidate document can be computed by taking the product of each

word's probability in the new document given the existing document.

$$P(q|\theta_u) = \prod_{w \in q} P(w|\theta_{\mathrm{u}})^{n(w,q)} \tag{2.3}$$

where θ_{u} denotes existing documents, $P(w|\theta_{\mathrm{u}})$ is the probability of generating word w from existing document θ_{u} and $n(w, q)$ is the number of times word w appears in the new document q.
More details for the implementation of the language models are presented in Chapter 3.

2.4.2 Topic Models

Since traditional vector space models suffer from the dimensionality problem, a better representation for text data has been one of the main issues for the data mining community. Several pre-processing methods have been applied to reduce dimensionality of traditional vector space models; these include removing stop words and stemming. But these approaches have been blamed for other reasons as well. For example, they do not take into account synonymic and polysemic relations of words.

To tackle these problems Probabilistic Models have been proposed. Probabilistic Models assume that data is generated through a random process. The statistical inference procedures infer the structure of the assumed random process using the observed data.

Latent Dirichlet allocation (LDA) [6], probabilistic latent semantic indexing (PLSI) [15] , hierarchical LDA (hLDA) and hierarchical Dirichlet processes (HDP) [33] are some examples of the well-known methods in this category. Since the latent variables in these models are mostly topics in the text corpus, these models are often referred to as "topic models".

Latent Dirichlet Allocation

The LDA model [6] is a three-level hierarchical Bayesian model and has been used extensively for modelling text corpora. This model assumes that documents are sampled from a random mixture over latent topics and each topic is assumed to be a distribution over words. In the LDA model, a prior probability is considered as the mixture coefficient which is assumed to be random. To generate a document, the LDA

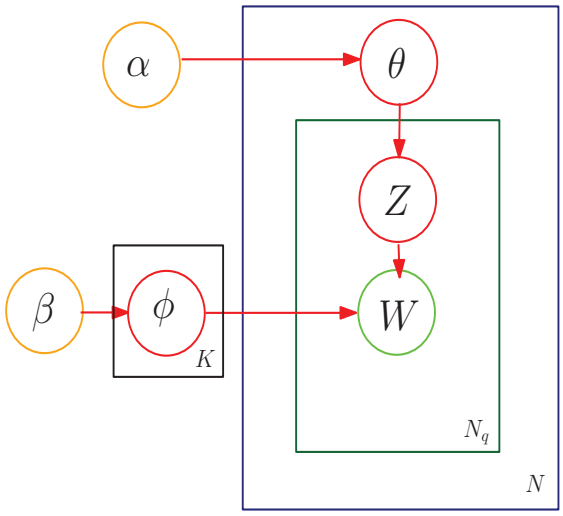


Figure 2.1: Graphical model representation of the Latent Dirichlet Allocation Model using plate notation.

model assumes that for each document a distribution over latent topics is sampled from a Dirichlet distribution. In the next step, for each word in the document a single topic is chosen according to this topic distribution. Finally, each word is sampled from a multinomial distribution over words specific to the sampled topic. The graphical model corresponding to the generative model of the LDA model is shown in Fig. 2.1.

To represent the graphical model for the LDA model, Plate notation has been used. Instead of representing each variable by an individual node, nodes with similar structural dependencies are illustrated and surrounded with a box called a plate, labelled by a number N that indicates N number of nodes of this type exists.

The generative process for each document is as follows:

1. Choose $N_{\mathrm{q}} \sim Poisson(\epsilon)$: Number of words in document d

2. Pick $\Theta_{\mathrm{d}} \sim Dir(\alpha)$

3. Repeat the process for each of the N_{q} words w_{dn}.

   (a) Choose a topic $z_{dn} \sim Multinomial(\theta_{\mathrm{d}})$

   (b) Pick a word w_{dn} from $P(w_{dn}|z_{dn}, \phi)$

In the LDA model, the probability of choosing term t for word n is equal to:

$$P(w_{dn} = t|\phi, \theta_d) = \sum_{k=1}^{K} P(w_{dn} = t|\phi_k)P(z_{dn} = k|\theta_d) \tag{2.4}$$

and the complete data-likelihood of document d is:

$$P(\overrightarrow{w_d}, \overrightarrow{z_d}, \theta_d, \phi|\alpha, \beta) = \prod_{n=1}^{N_q} \sum_{k=1}^{K} P(w_{dn} = t|\phi_k)(P(z_{dn} = k|\theta_d))P(\theta_d|\alpha)P(\phi|\beta) \tag{2.5}$$

where $\overrightarrow{w_d}$ shows the vector of all the words in document d and $\overrightarrow{z_d}$ denotes the corresponding topics of these words. At the next step, we can integrate out parameters θ and ϕ and sum over z_{dn}:

$$\begin{aligned} P(w_{\mathrm{d}}|\alpha, \beta) &= \int\int P(\theta_d|\alpha)P(\phi|\beta) \prod_{n=1}^{N_q} \sum_{z_{dn}} P(w_{dn} = t|\phi_k)P(z_{dn} = k|\theta_d)\mathrm{d}\phi\mathrm{d}\theta_d \\ &= \int\int P(\theta_d|\alpha)P(\phi|\beta) \prod_{n=1}^{N_q} P(w_{dn} = t|\phi_k)\mathrm{d}\phi\mathrm{d}\theta_d \end{aligned} \tag{2.6}$$

Limitations of the LDA Model

Topic models are more precise for specifying the structure of text compared to traditional models because they use a generative model framework. The Latent Dirichlet Allocation has overcome some of the traditional models' problems in terms of extracting more information from data, but it still is too simple and has some limitations. In the following chapters, we will propose models which improve these limitations.

One of the limitations of LDA is that it assumes that paragraphs and sentences within documents are exchangeable. In the LDA model, exchangeability means paragraphs are conditionally independent and generation of each word in a document is independent of all other words in the document, but both of these assumptions are not realistic in many documents. For example, in CQA document collections, the order of questions answered by a user and the date of each post reflects how interests of the users can change over the time. Therefore, the exchangeability of words may not be a proper assumption in that collection.

Another limitation is that the LDA model is not able to capture correlations between topics. In this model, topics are extracted from a Dirichlet distribution and are independent.

Therefore, we need models that include both the distribution over words and distribution over topics. By this assumption we have topics that are mixture of other topics. In the next chapter, we will introduce two models that have properties that are lacking in the LDA model.

Chapter 3

Methodology

3.1 Introduction

In the previous chapter, we introduced the LDA model and distinguished some of its strengths compared to traditional information retrieval approaches. We also mentioned that the LDA model assumes that the textual data is unstructured; it groups the whole content of a document under a single topic distribution. Therefore, it may not be able to capture certain properties of text. Our goal is to use models that take advantage of the structure and additional information of text data to tackle some of the limitations of the LDA model.

In this chapter, we first introduce a model to exploit the interest information of users for enriching the LDA model. Later on, we use an existing model that recognizes the boundary information of text documents to assign a different topic distribution to each portion of text instead of assigning a single topic distribution to the whole document.

The intuition behind these models is that the words in a sentence, paragraph of a text, or in a question of a user profile have stronger correlation with other words in that sentence, paragraph, or question compared to other words in the document. There are also intra-part relationships between different paragraphs of a text document as well as different questions of a user profile.

The models that we discuss in this chapter take advantage of the inter-part and intra-part correlations.

3.2 Problem Statement

Given a new question q, we need to return a ranked list of users u_1, u_2,...,u_n who are best suited to answer q. The probability of a user u, being the answerer for the

question q is $P(u|q)$ and, by Bayes theorem, can be expressed as::

$$P(u|q) = \frac{P(u)P(q|u)}{P(q)} \tag{3.1}$$

where $P(q)$ is the probability of question q and we assume it is uniform for all the test questions. $P(u)$ is the prior probability of user u which can be approximated by specific information such as user activity derived from the dataset. In this study, our objective is to compute the probability $P(q|u)$ that captures the expertise of user u on question q. This probability model was first introduced by [3].

The optimization task in this problem is to maximize the probability assigned to the best suited expert in each model. All of the different methods used for estimating attempt to maximize this probability. In the word-based model, the probability of generating a question from different user profiles is maximized, when there is considerable word overlap between user profile and question. In topic models, on the other hand, this probability is maximized when there is enough similarity between distribution over topics in question and user profiles. Therefore, our goal is to optimize this probability.

In the dataset used in this research, each question has three parts: question tags, question title, and question body. Question tags are the tags assigned by users who posted the questions. Question title is a short description of the question. The detailed description is given in the question body. The main challenge is representing the questions. Additionally, the expertise and interest of a user should be modelled by taking advantage of the activity history of the user.

3.3 Modeling Expert Search

In the Community Question Answering Services, answerers usually choose a category that they are more interested in and then pick a question from that category. Therefore, user interests can be inferred from answering history. In this section, we explore different methods for ranking users based on their interests. These methods can be divided into two main categories: word-based methods and topic models. In the first category, we model user interest by using TF-IDF and language model. In general, word-based methods use a smoothed distribution to estimate the likelihood of a query

in a given collection of documents. Topic models have an additional representational level. Documents in these models are a mixture of topics and topics are mixtures of words.

3.3.1 User Persona Model

In the LDA model, topics are derived from a Dirichlet distribution which considers the topics to be independent. This means that the presence of a topic is not correlated to the presence of another topic [5].

Several models, including the Hierarchical Dirichlet Process Model (HDP) [34], Correlated Topic Model (CTM) [5], Pachinko Allocation Model (PAM) [19] and Latent Dirichlet Co-Clustering model [32] have been introduced to capture the correlation between topics.

In this part, we propose a generative model for text documents based on the LDA model. This model is designed to capture the correlation between topics through a higher level of Personas. This model assigns a persona to each question in the user profile, which makes the model more conscious to the locations of words in the question and also more focused on each single question. The pupose behind assigning a persona to a question and also a distribution of personas to a user is to capture the intra-question dependency of user profile.

Model Definition

We define the following terms:

- The building blocks of our data are words selected from a vocabulary indexed by $\{1,...,V\}$.
- A document is a mixture of words $w = (w_1,...,w_{\mathrm{N}})$ where N is the total number of words in the document.
- A corpus is a collection of M documents shown as $\{d_1,...,d_{\mathrm{M}}\}$.

The LDA model represents each document as a mixture of latent topics where each topic is a distribution over words. We extend this idea by assuming that each user profile is a mixture of personas that reflect the user's interests and represent higher

level topics. A persona generally groups different topics in the lower level of document generation where each of the topics is a distribution over words. Each question in the user profile then can be modelled by the LDA model.

The intuition behind this model is that user profiles are composed of single high-level topic questions. Generally speaking, each of these questions conveys a single concept. These topics or concepts that define the theme of a user profile can be used to predict user interests among a handful number of interests. The order of words in each question is assumed to have little effect on the concept which an individual question conveys; the assumption of the "bag-of-words" in this case is more realistic than the one in LDA.

User Persona Model (UPM) tries to model each question based on its content similar to other probabilistic topic models and then use these learned topics to represent persona distribution of user profile. This means that each persona in the user profile is considered as a mixture of topics.

UPM is designed to model the interest of users. The number of questions in each user profile is much fewer compared to the number of words in it. Each question in user profile is assigned to a persona. These assignments show how the user profile is divided between different personas and will give the main focus of the users interests.

The graphical model of the UPM model is shown in Fig. 3.1(a). In this graphical notation, the only observed variable is w and the rest are latent.
The generative process for each document d in a corpus D is:

1. Choose $N_Q \sim Poisson(\gamma)$: number of questions in the user profile

2. Choose $\pi_u \sim Dir(\alpha_\pi, P)$: mixture components of personas, where P is the total number of personas.

3. For each question q of user u:

   (a) Choose a persona for the question q: $Y_q^u \sim Multinomial(\pi_u)$

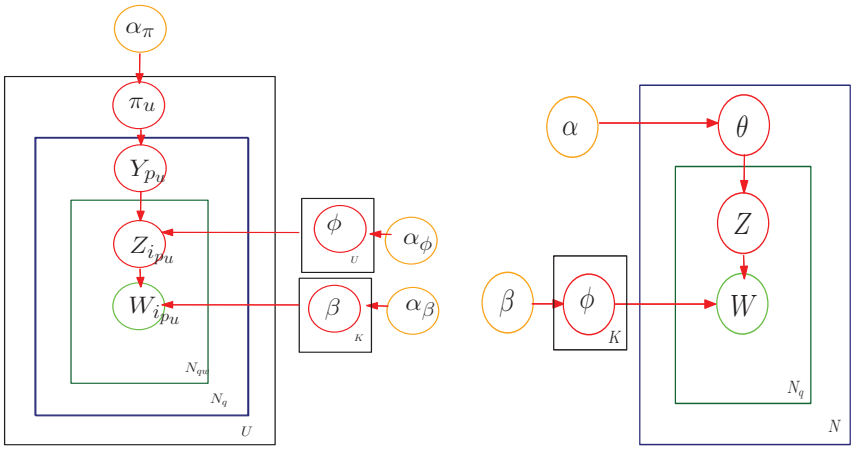
   (b) Choose $NW_q^u \sim Poisson(\epsilon)$: number of words in the question q of user u

   (c) For each of the q words W_q^u:

      i. Choose a topic $z_{W_{q_u}} \sim Multinomial(\beta)$

ii. Choose a word w_{sn} from $P(w_{sn}|z_{W_{qu}}, \beta)$ a multinomial probability conditioned on the topic $z_{W_{qu}}$

In the UPM model, we have assumed that the number of topics and personas are known and fixed. We also model the word probabilities conditioned on the topics by



(a) UPM Model (b) LDA Model

Figure 3.1: The number of topics is assumed to be given. The only observed random variable in this model is W.

a $K \times V$ matrix which is estimated through the learning process.

Note that in the generative process, the N_q variable is independent of all other variables (θ, z, π, and y). π represents the mixing portion of persona in a user profile. It shows the parameters of the P-dimensional multinomial distribution that samples of personas are drawn from. Y_{Q_u} is a sample from the Dirichlet distribution and specifies the mixing proportion of topics in a question q of user u. This mixing proportion depends on the persona from which the current question is generated. The UPM model assumes that each persona is a mixture of several topics and this fact is modelled by using the α_β hyper parameter.

Given the parameters α_π,α_ϕ and α_β, the joint distribution of a topic mixture ϕ, persona mixture π, a set of K topics z, a set of P personas Y and a set of words w

is:

$$P(w, Z, Y, \pi, \phi, \beta | \alpha_\pi, \alpha_\phi, \alpha_\beta)$$
$$= \prod_{u=1}^{U} P(\pi_u|\alpha_\pi) \prod_{q=1}^{N_Q} P(Y_q^u|\pi) \prod_{n=1}^{N_{WQ}} P(Z_{qn}^u|Y_q^u, \phi) P(w_{qn}^u|Z_{qn}^u, \beta) \prod_{p=1}^{P} P(\phi|\alpha_\phi) \prod_{k=1}^{K} P(\beta|\alpha_\beta) \tag{3.2}$$

Variables Z_{qn}^u and Y_q^u are vectors that one if their components is equal to one and all other components are zero. The component equal to one in Z_{qn}^u represents the topic corresponding to word n and the one in Y_q^u represents the persona assigned to question q.

A complete set of derivations for generating user profiles has been presented in Appendix A.

Inference and Parameter Estimation

The inference problem refers to the process of computing the posterior distribution of hidden variables given the input variables $\alpha_\pi, \alpha_\phi, \alpha_\beta$ and observed variable w:

$$P(Z, Y, \pi, \phi, \beta | w, \alpha_\pi, \alpha_\phi, \alpha_\beta) = \frac{P(Z, Y, \pi, \phi, \beta, w | \alpha_\pi, \alpha_\phi, \alpha_\beta)}{p(w|\alpha_\pi, \alpha_\phi, \alpha_\beta)} \tag{3.3}$$

which generally is hard to compute.

Performing the exact inference on topic models is not possible practically. However, some approximation methods have been proposed to perform the inference: variational methods [6] , Gibbs sampling [30] and expectation propagation [13]. To perform the inference we choose a method in which some of the hidden parameters can be integrated out. To simplify the sampling we use conjugate priors in our model.

We use Gibbs sampling to draw a sample from the high dimensional distribution. Gibbs sampling is an approximate method and is an iterative algorithm. In each iteration of the algorithm, a subset of variables is sampled from their conditional distribution conditioned on the current values of all other variables. This process is continued until the sample values approach the desired distribution. In our model, we want to sample the distribution of topics and distribution of personas. Since these distributions are hard to sample, we assume that the topic assignments for all other words in all user profiles are known except for the word that we are going to sample

from the conditional distribution. Also, when we sample from the distribution of a question, we assume that all the persona assignments of all other questions are known.

For each word in the corpus, the Gibbs sampling algorithm estimates the probability of assigning that word to a topic assuming that the topic assignments of all other words are known. Also, for each question, the Gibbs sampling algorithm estimates the probability of assigning a question to a persona given the assignments of all other questions to their personas. In the first Appendix, the conditional distributions of topics and personas are explained in detail.

The initial iteration of the Gibbs sampling algorithm begins with assigning a random topic to each word and a random persona to each question. Because of the random initialization, a number of initial samples have to be ignored (burn-in period). The samples start to approach the desired distribution after the burn-in period. The next step involves choosing a number of samples and saving them as representative samples from the distribution.

However, the Gibbs sampling algorithm of the UPM model never reached the target values and it failed to estimate the topic and persona distributions. After a number of iterations, the probabilities of a subset of personas suddenly start to decrease and the algorithm rounds them toward zero. The pseudo code for the Gibbs sampling process for the UPM model is shown in Algorithm 1 and associated symbols are presented in Table 3.1. All the details of the UPM model are available in Appendix A for further studies.

Many test experiments were performed on the model to address the failure of the UPM model. Surprisingly, the model did not show any sign of failure on small datasets but experiments on the large datasets were time consuming and on our limited time, we were not able to undertake such large scale experiments. Therefore, we decided to take advantage of a similar existing model that most possessed the specific characteristics we desired. In the next section, we will briefly explain this model.

3.3.2 Segmented Topic Model

LDA is informative about the content of user profiles. But it does not take advantage of the structure of profiles. Each profile is composed of questions where each question

Algorithm 1 UPM Gibbs Sampling Algorithm

Input: $\alpha_\pi, \alpha_\phi, \alpha_\beta, MaxtIteration, Corpus, K, P$

Output: topic and persona assignments for all the words and questions in the corpus

{Add nodes from $G - R$ to E as long as the total cost improves}

Initialization
Randomly initialize the topic and persona assignments for all the words and questions
Compute NUP for all values of $p \in \{1, ..., P\}$ and all the user profiles
Compute NPK for all values of $k \in \{1, ..., K\}$ and all the personas assigned to all the questions in the corpus
Compute NKV for all values of $k \in \{1, ..., K\}$ and all the words
for $iter \leftarrow$ to $MaxIteration$ **do**
  for each profile u in the corpus **do**
    for each question q in u **do**
      Exclude q and its assigned personas p from NUP and NPK
      $newP$ = sample new persona for question q using Eq. A7
      update NUP and NPK using the new persona $newP$ for question q
      for each word w in question q of profile u **do**
        Exclude word w and its assigned topic k from NPK and NKV
        $newK$ = sample new topic for word w using Eq. A6
        update NPK and NKV using the new topic $newK$ assigned to word w
      end for
    end for
  end for
end for

Table 3.1: A brief description of symbols used for the UPM model

Notation	Description
K	Number of topics
P	Number of personas
U	Number of profiles
N_{q}	Number of questions in profile u
$N_{\mathrm{q,w}}$	Number of words in question q, profile u
V	Size of vocabulary
y	Persona variable for the corpus
z	Topic variable for the corpus
α_π	Parameter of the Dirichlet prior on personas
α_ϕ	Parameter of multinomial distribution of topics conditioned on personas
α_ϕ	Parameter of multinomial distribution of words conditioned on topics
π	Mixing proportion of personas in user profile
ϕ	Mixing proportion of topics in personas
β	Mixing proportion of words in topics
$w_{u,q,v}$	Word in profile u, question q, at position v
$z_{u,q,v}$	Topic for word in profile u, question q, at position v
NUP	A matrix shows number of times that persona p assigned to user u
NPK	A matrix shows number of times that topic k assigned to a persona p
NKV	A matrix shows number of times that word w assigned to topic k

contains sentences. The shared topics between questions can be extracted from the structure of profiles.

Segmented Topic Model (STM) introduced by Lan Du et al. [11] is a topic model that discovers the hierarchical structure of topics by using the two-parameter Poisson Dirichlet process [27]. A four-level probabilistic model, STM contains two levels of topic proportions. Instead of grouping all the questions of a user under a single topic distribution, it allows each question to have a different and separate distribution over the topics. This can lead to more realistic modelling of expertise.

In the STM model, words are the basic element of the data represented by 1,..,W. Each question q is considered as a segment that contains $N_{\mathrm{q,w}}$ words. A user profile is considered as a document that contains questions (segments). A corpus is a collection of profiles. The complete list of notations is shown in Table 3.2.

Each profile u is a mixture of latent topics denoted by probability vector μ_{u}; each question is also a mixture on the same space of latent topics and is drawn from a probability vector $\nu_{\mathrm{u,q}}$ for question q of profile u. The expertise set of a user, the main topics of each question in the profile, and the correlation between each profile and its questions are modelled by these distributions over topics μ and ν.

The generative process of STM for a profile u is as follows:

1. Pick $\mu_{\mathrm{u}} \sim \mathrm{Dirichlet}(\alpha)$.

2. For each question q draw $\nu_{\mathrm{u,q}} \sim \mathrm{PDP(a,b,}\mu_{\mathrm{u}})$.

3. For each word $w_{u,q,v}$ choose a topic from $z_{u,q,v} \sim$ discrete $(\nu_{\mathrm{u,q}})$.

4. Select a word from $w_{u,q,v} \sim$ discrete$(\phi_{z_{u,q,v}})$.

The graphical representation of STM is shown in Fig. 3.2(a). The number of topics is assumed to be given. The only observed random variable in this model is W. More details about the STM model is presented in Appendix A.

The hyper parameters in topic models have a great influence on representing the topics in user profile. By choosing low values for these parameters, we prevent the sparseness of Dirichlet distribution and force the models to give more importance to the frequent topics and active users. However, when we choose higher values for hyper parameter, we allow the topics to belong to wider range and cover the infrequent

Table 3.2: A brief description of symbols used for the STM model

Notation	Description
K	Number of topics
U	Number of profiles
N_{q}	Number of questions in profile u
$N_{\mathrm{q,w}}$	Number of words in question q, profile u
V	Size of vocabulary
α	Prior distribution for profile topic distribution
μ_{u}	Profile topic probabilities for profile u
$\nu_{\mathrm{u,q}}$	Question topic probabilities for user u, question q
ϕ	Words probability matrix
γ	Dirichlet prior for ϕ
$w_{u,q,v}$	Word in profile u, question q, at position v
$z_{u,q,v}$	Topic for word in profile u, question q, at position v

topics and quiet users. However, in the later case, the models may loose the focus on the important datas and not associate enough attention to the frequent users.

3.3.3 TF-IDF

In the previous chapter, we introduced TF-IDF. In this section, we present the details of using TF-IDF for the expert retrieval task.

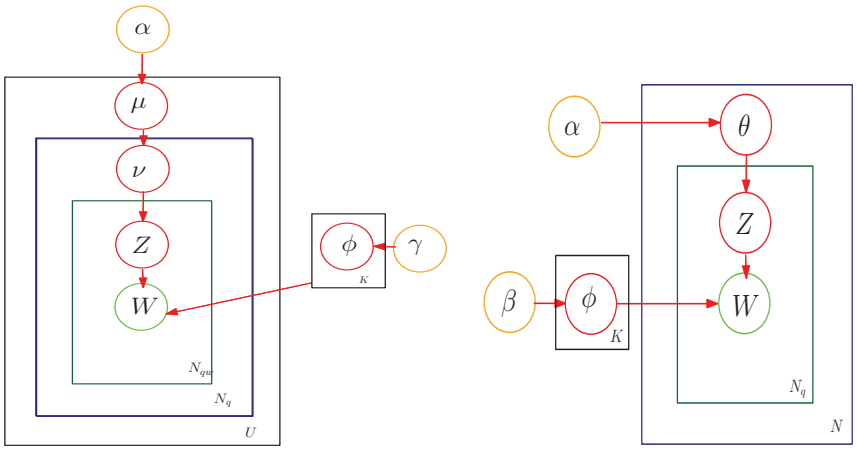
Given a test question q composed of a set of words, we represent the test question and each user profile as vectors of their tf-idf weights and then calculate the Cosine Similarity between each user profile and the question vector:

$$s(u,q) = \frac{\sum_w tfidf(u,w)tfidf(q,w)}{\sqrt{\sum_w tfidf(u,w)^2}\sqrt{\sum_w tfidf(q,w)^2}} \tag{3.4}$$

where $tfidf(q,w)$ is the tf-idf weight of word w in q, and $tfidf(u,w)$ is the tf-idf weight of w in the profile of user u.

3.3.4 Language Model

To use the language model in the expert retrieval task, a multinomial probability distribution over words in the vocabulary is used to represent a candidate user. A new question is represented by a set of words q= $\{w_1,w_2,...,w_{\mathrm{N}}\}$ where w_{i} is a non-stop word and each question is assumed to be generated independently. Therefore,



(a) STM Model (b) LDA Model

Figure 3.2: Graphical model representation of the LDA Model and STM Model. The only observed variable in these models is W

the probability of a question being generated by a candidate user can be computed by taking the product of each word's probability in the question given the user profile.

$$P(q|u) = \prod_{w} P(w|\theta_{\mathrm{u}})^{n(w,q)} \tag{3.5}$$

where θ_{u} denotes the user profile for user u, $P(w|\theta_{\mathrm{u}})$ is the probability of generating word w from user profile θ_{u} and $n(w,q)$ is the number of times word w appears in question q. Since many words in the vocabulary will not appear in a given user profile, and $P(w|\theta_{\mathrm{u}})$ will be zero for such w, we need to use a smoothing method on the $P(w|\theta_{\mathrm{u}})$. By doing so, we can avoid zero probability for unseen words [36]. We apply Dirichlet smoothing method for $P(w|\theta_{\mathrm{u}})$:

$$P_{LM}(w|\theta_{\mathrm{u}}) = \lambda P(w|\theta_{\mathrm{u}}) + (1-\lambda)P(w) \tag{3.6}$$

where $P(w)$ denotes the background language model built on the entire collection Q and $\lambda \in [0,1]$ is a coefficient to control the influence of the background model and is defined as:

$$\lambda = \frac{\sum_{w \in \theta_{\mathrm{u}}} tf(w, \theta_{\mathrm{u}})}{\sum_{w \in \theta_{\mathrm{u}}} tf(w, \theta_{\mathrm{u}}) + \mu} \tag{3.7}$$

where $tf(w, \theta_{\mathrm{u}})$ is the frequency of w in the profile of u and parameter μ is set to 1000 in the experiments.

The background model $P(w)$ can be computed through a maximum likelihood estimation:

$$P(w) = \frac{n(w, Q)}{|Q|} \tag{3.8}$$

where $n(w, Q)$ denotes the frequency of words w being in the collection Q and $|Q|$ is the total number of words in the collection.

Chapter 4

Experimental Study

The experimental dataset is based on a snapshot of the community based question answering site Stackoverflow[1]. It features questions and answers on a wide range of topics in computer programming.

4.1 Stackoverflow

The purpose of this section is to provide readers with the necessary background to understand the main characteristics and the question answering mechanism of the Stackoverflow website.
Stackoverflow is a question answering service organized with a user-defined taxonomy of topics. As the organizers mentioned in the "about section" of the website, Stackoverflow is an original way to synchronize the aspects of Wikis, Blogs, Forums, and Digg/Reddit.

Questions and answers are posted within categories. The writer of a question should specify the category of the question by assigning a keyword or tag for it. There are approximately three thousand different tags in Stackoverflow. The categories cover a range of different topics in computer programming and attract users from a wide variety of fields. Stackoverflow participants can thus save time in their quest for information because they can get an answer quickly or find what they are looking for among the existing questions and answers.

Questions are the central elements of Stackoverflow. The life cycle of a question starts in an open state where it receives several answers. Then, at some point, a best answer is selected either by the user who posted the question or by other users via a voting procedure. The question will be considered closed once a best answer is chosen.

Note that Stackoverflow participants do not limit their activity to asking and

[1] http://stackoverflow.com/

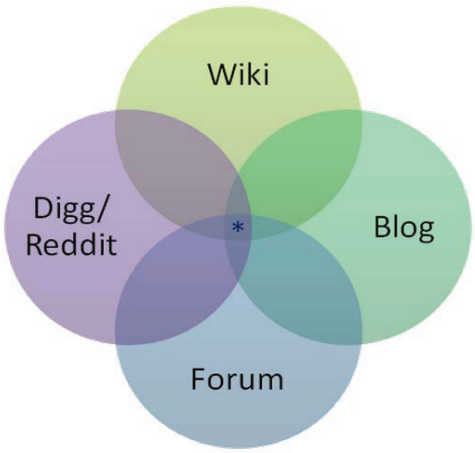


Figure 4.1: Stack Overflow is that tiny asterisk in the middle[a]

[a]http://stackoverflow.com/about

Table 4.1: Data Statistics of Stackoverflow, the statistics belong to January 2011 data dump

Questions	1,188,585
Answers	2,939,767
Users	440,672

answering questions. They are also allowed to participate in regulating the system by voting and editing questions and answers. Users can mark interesting questions and evaluate the answers by voting for the best answers. Stackoverflow presents additional data, i.e. each user has a reputation and badge. This type of information can be used as ground truth for the performance evaluation. Stackoverflow releases its data dump every two month. The statistics of the January 2011 data dump is shown in Table 4.1.

4.2 Dataset

To conduct experiments, we select a representative subset of the dataset. Tags are the only elements that categorize different topics. However, they belong to a very diverse topic set. Therefore, we need to create a subset of the dataset that exhibits the same properties as the original one. Frequency of the top 2000 tags from posts with more than 2 answers is shown in Fig. 4.2. We use this diagram to extract the most frequent tags. The pair-wise frequency of the top 2000 tags from posts with more

Table 4.2: 21 selected tags for the training set

Frequently co-occur	partially co-occur	Rarely co-occur
C#	Bach	Django
SQL	Python	css
Linux	SQL-server	Ruby
Windows	Delphi	Ruby-on-rails
Java	Web-development	WPF
C	.net	iphone
Homework	Java script	Android

Table 4.3: Data Statistics. Numbers in parentheses show candidate best answerers for expert prediction

Questions	Askers	Best Answerers
369440(123933)	186027	22027(1845)

than 2 answers is presented in Fig. 4.3. By using these two diagrams, we examined tag frequency and tag co-occurrence statistics, and manually selected a total of 21 tags. This subset was chosen such that a similar tag distribution as the original data collection was maintained. Selected tags mostly belong to three categories: (i) tags that are highly frequent and mainly co-occur with other tags, (ii) tags that are frequent but never co-occur with other tags, and (iii) tags that sometimes co-occur with other tags. These tags are shown in Table 4.2.

Stackoverflow website is crawled and 118510 resolved questions and answers between Jan 2008 and Jan 2009, tagged with one or more of the 21 selected tags are picked. This dataset is publicly available for research purposes[2].

The questions are resolved, thus each question has a best answer corresponding to one best answerer. Some statistics for this dataset are shown in Table 4.3. Numbers in parentheses are related to candidate best answerers. For the best answerer prediction, those users who have answered at least N best answers are considered (in this work $N = 20$). As seen in Table 4.3, in the selected dataset 22027 users have given at least one best answer, while only 1845 of users wrote at least 20 best answers. Those 1845 users are very important to the question answering community. They constitute 0.5% of all users, but have answered 35% of all answered questions.

All the questions are stemmed and the stop-words have been removed using Mallet

[2]http://web.cs.dal.ca/~riahi/

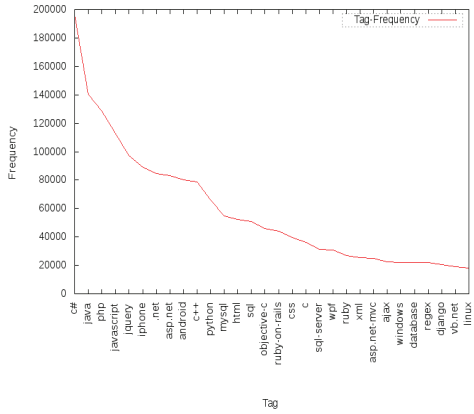


Figure 4.2: Distribution of the most frequent tags in Stackoverflow

toolbox [3] [22]. Words that appear less than 5 times in the corpus are also ignored. All the questions answered by the best answerers between January and February 2009 are extracted to build the test dataset. As a result, there are 5128 test questions in our test dataset. Note that some best answerers give no best answer to questions within this period. Therefore, there is no test question assigned to them in our test dataset. However, they are still candidate users for best answerer prediction for a given test question.

We implemented the models and ran them on one of the Acenet [4] servers with AMD core and 2 Quad CPUs (2.4 GHz) and 3 Dual core CPUs. The codes were not multi-threaded. The running time is approximately 3 hours for TF-IDF and 8 hours for language model. The running time on the dataset with 100 topics and 3000 iterations is approximately 10 hours for LDA and 120 hours for STM.

4.3 Evaluation

Evaluation of the quality of the resulting ranked list of best users is a difficult task. Only users who have already answered a particular question are ranked in [29]. This will be useful in choosing the best answer for the question but will not help in directing a new question to potential best experts. However, in [20], all the users in the corpus are ranked for the given question instead of ranking only those users who have answered the question. In our work, we used the second method and ranked the users

[3] http://mallet.cs.umass.edu/
[4] http://www.ace-net.ca/wiki/ACEnet

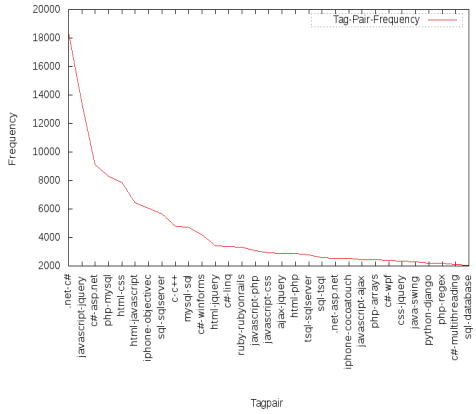


Figure 4.3: Distribution of the most frequent co-occurring tags in Stackoverflow

according to the four methods mentioned previously. If a model could find the actual best answerer of the questions among the top N predicted users, then, the prediction is considered successful. This method of measuring the quality of ranking is called success at N ($S@N$).

For $S@N$, if the best answerer for a test question is among the top N returned users, then the value of $S@N$ is the reciprocal rank of that user, otherwise the $S@N$ value is 0. The value of $S@N$ of all the test questions is the average $S@N$ value of the whole test set. Therefore, a large value for $S@N$ means better performance. In the best case, when the best answerers for all the test questions are ranked number one by a method, $S@N$ will be 1.

In topic models, hyper-parameters could play an important role. For the LDA model, a range of values between 0.01 and 0.05 for parameter α were explored. We also tried different settings of a and b for the STM model and eventually used $a = 0.2$ and $b = 10$ for all the experiments.

The results of S@1 for different number of topics is shown in Fig. 4.4. Y axis represents the S@1 values and X axis shows number of topics. Number of topics is a parameter of the topic models we have used. Performance for TF-IDF and language model is independent of this parameter and therefore, it stays the same as the number of topics changes for our topic models.

The results of predicting best answerers comparing four different methods are presented in Table 4.4. As it can be seen topic models exhibit much better performance compared to the two traditional information retrieval approaches. As we expected,

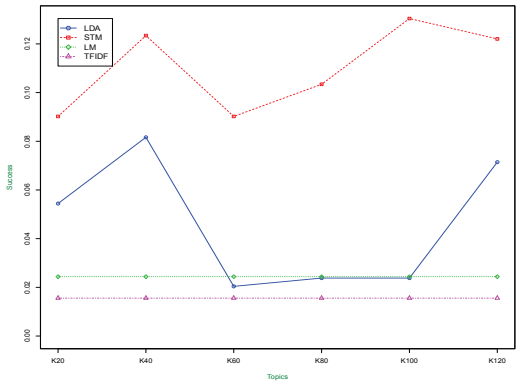


Figure 4.4: Results of best answerers prediction. Y axis shows S@1 values and X axis represents number of topics

the STM model consistently performs better than the LDA model which indicates that taking advantage of the structure of profiles is important in retrieving the answerers. In general, semantic based methods seem to be more accurate in predicting the best answerer in our corpus.

Some examples of topics extracted from user profiles using the STM and LDA models are shown in Table 4.5 and Table 4.6 respectively. The purpose of these two tables is to intuitively demonstrate that the topics extracted using the STM model are superior compared to the ones extracted using the LDA model. Comprehensive user studies are required to verify our intuitive conclusion. The first 5 columns in Table 4.5 are some examples of topics extracted by the STM model, which the LDA model failed to properly detect. For example, the second column of Table 4.5 shows the computer graphics topic discovered by the STM model. The corresponding topic discovered by the LDA model is shown in the second column of Table 4.6. Comparison of these two columns suggests that the topic discovered by the STM model is more coherent compared to the corresponding topic discovered by the LDA model. It might seem unfair that we have compared the good quality topics discovered by the STM model against the corresponding topics found by the LDA model. Therefore, the last 5 columns in both tables show topics that were of high quality discovered by the LDA model and their corresponding topics found by the STM model. Comparing these sets of topics indicates that wherever the LDA model is performing well, the STM model can match its performance in terms of topic quality.

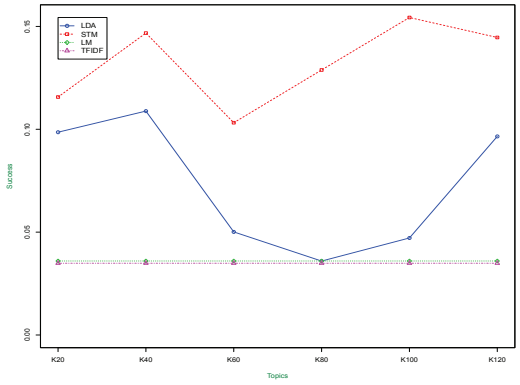


Figure 4.5: Results of best answerers prediction. Y axis shows S@5 values and X axis represents number of topics

Table 4.4: Results of best answerers prediction for S@N

Method	S@1	S@2	S@3	S@4	S@5
LM	0.0243	0.0304	0.0304	0.0335	0.0359
TF-IDF	0.0155	0.0272	0.0298	0.0317	0.0348
LDA	0.0578	0.0765	0.0810	0.0836	0.0856
STM	0.1034	0.1051	0.1192	0.1200	0.1267

Table 4.5: The first 10 most probable words for 10 different topics in STM

Topic 1	Topic 2	Topic 3	Topic 4	Topic 5	Topic 6	Topic 7	Topic 8	Topic 9	Topic 10
size	colour	servlet	server	email	game	iphone	os	table	thread
bit	draw	request	client	session	player	app	linux	query	lock
heap	graphic	response	connect	send	point	device	windows	select	run
space	png	message	socket	mail	rotate	application	mac	row	wait
cpu	img	web	port	attach	graph	mac	system	column	synchronize
machine	background	server	send	subject	video	apple	platform	join	start
allocation	jpg	redirect	remote	receive	matrix	sdk	virtual	order	call
usage	rectangle	session	network	queue	scale	opengl	pc	sql	process
limit	height	tomcat	host	address	board	os	machine	result	block
run	circle	http	ip	body	math	video	bit	mysql	sleep

Table 4.6: The first 10 most probable words for 10 different topics in LDA

Topic 1	Topic 2	Topic 3	Topic 4	Topic 5	Topic 6	Topic 7	Topic 8	Topic 9	Topic 10
size	colour	servlet	server	email	game	iphone	os	table	thread
object	graphical	public	file	user	play	app	windows	id	run
memory	insert	string	code	post	develop	data	install	select	lock
class	collect	handle	string	view	program	animation	application	query	object
code	google	id	http	format	card	view	mac	key	wait
error	value	class	content	comment	sound	image	local	row	call
snapshot	jsp	persist	workspace	valid	video	application	framework	group	time
message	file	tag	line	page	audio	object	library	method	synchronize
profile	database	google	ibm	update	code	sdk	process	group	method
row	code	message	test	action	book	video	win	type	queue

Chapter 5

Conclusion

Topic models discover concepts from text corpora that traditional models usually fail to discover. These models can be extended to more complicated models which can extract more aspects of the textual content. The simple topic models usually ignore this information and treat text data as a bag of words to reduce computational complexity.

In this research, we presented a series of models to extract the structure information from the text data to improve the quality of results of expert finding approaches in community question answering services.

Routing new questions to the right group of experts is an important problem in Community Question Answering systems. A solution to this problem provides users with high quality answers within a reasonable time. It also presents questions to the experts matching their expertise.

For experts in our dataset, we build profiles based on their answering history. These profiles are then used in comparison with a newly posted question. Two statistical topic models are used along with two more traditional approaches. Latent Dirichlet Allocation model, the TF-IDF, and the language model assume that a user profile is a single text unit comprising all questions answered by the user. The Segmented Topic Model, on the other hand, recognizes individual questions as independent units of text. Our results indicate that the LDA model outperforms TF-IDF and language model in retrieving a candidate set of best experts for a question. The STM model performs considerably better than the LDA model, suggesting that the simple structural information used in the model helps produce better results. Our results suggest that statistical topic models can be considered as suitable replacements for more traditional methods in expert recommendation.

The topic models in the expert finding task can be applied on the online data as well. These models will be trained once to find the topic assignments of different users.

Then, for each new question in the system, the already trained model has been used each time. Therefore, there is no need to repeat training from the scratch if only small part of data has changed. In the current implementation of the expert recommender, there is no difference between recent and old posts. In future modifications, it would be interesting to see the effect of time. Basically the interests of the users change over time and recent questions answered by a user more accurately reflect the interests of the user than the old questions. However, the models we used in this thesis, assumed that topics do not change over time. To address this problem, we can take advantage of dynamic topic models to give more importance to the new questions. We can also use a weighting approach to give more weight to the recent questions both in topic models and traditional models.

Community Question Answering websites produce other types of metadata for the posted questions and answers such as score, favourite count, and last edit date. Moreover, user information often contains metadata information such as badges and reputation. Using this additional information may help improve the performance of an expert recommendation system. Statistical topic models can be extended to model additional observed variables. Encouraged by the performance improvement for the STM model, we are planning to take advantage of this information in our future work by including extended list of variables.

Bibliography

[1] Lada A. Adamic, Jun Zhang, Eytan Bakshy, and Mark S. Ackerman. Knowledge sharing and yahoo answers: everyone knows something. In *Proceedings of the 17th International Conference on World Wide Web*, WWW '08, pages 665–674. ACM, 2008.

[2] Ricardo A. Baeza-Yates and Berthier Ribeiro-Neto. *Modern Information Retrieval*. Addison-Wesley Longman Publishing Co., Inc., 1999.

[3] Krisztian Balog, Leif Azzopardi, and Maarten de Rijke. Formal models for expert finding in enterprise corpora. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '06, pages 43–50. ACM, 2006.

[4] Adam Berger, Rich Caruana, David Cohn, Dayne Freitag, and Vibhu Mittal. Bridging the lexical chasm: statistical approaches to answer-finding. In *Proceedings of the 23th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '00, pages 192–199. ACM, 2000.

[5] David M. Blei and John D. Lafferty. A correlated topic model of science. *AAS*, 1(1):17–35, 2007.

[6] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, 2003.

[7] Robin D. Burke, Kristian J. Hammond, Vladimir Kulyukin, Steven L. Lytinen, Noriko Tomuro, and Scott Schoenberg. Question answering from frequently-asked question files: Experiences with the faq finder system. Technical report, AI Magazine, 1996.

[8] Yunbo Cao, Huizhong Duan, Chin yew Lin, Yong Yu, and Hsiao wuen Hon. Recommending questions using the mdl-based tree cut model. In *Proceeding of the 17th International Conference on World Wide Web, (WWW 08*, pages 81–90. ACM Press, 2008.

[9] Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41:391–407, 1990.

[10] Byron Dom, Iris Eiron, Alex Cozzi, and Yi Zhang. Graph-based ranking algorithms for e-mail expertise analysis. In *Proceedings of the 8th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 42–48. ACM, 2003.

[11] Lan Du, Wray Buntine, and Huidong Jin. A segmented topic model based on the two-parameter poisson-dirichlet process. *Mach. Learn.*, 81:5–19, 2010.

[12] Yupeng Fu, Rongjing Xiang, Yiqun Liu, Min Zhang, and Shaoping Ma. A cdd-based formal model for expert finding. In *Proceedings of the Sixteenth ACM Conference on Information and Knowledge Management*, CIKM '07, pages 881–884. ACM, 2007.

[13] Thomas L Griffiths and Mark Steyvers. Finding scientific topics. 101:5228–5235, 2004.

[14] Jinwen Guo, Shengliang Xu, Shenghua Bao, and Yong Yu. Tapping on the potential of q&a community by recommending answer providers. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management*, CIKM '08, pages 921–930. ACM, 2008.

[15] Thomas Hofmann. Probabilistic latent semantic indexing. In *Proceedings of the 22th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '99, pages 50–57. ACM, 1999.

[16] Jiwoon Jeon, W. Bruce Croft, and Joon Ho Lee. Finding semantically similar questions based on their answers. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 617–618. ACM, 2005.

[17] Jiwoon Jeon, W. Bruce Croft, and Joon Ho Lee. Finding similar questions in large question and answer archives. In *Proceedings of the 14th ACM international conference on Information and knowledge management*, CIKM '05, pages 84–90. ACM, 2005.

[18] Jiwoon Jeon, W. Bruce Croft, Joon Ho Lee, and Soyeon Park. A framework to predict the quality of answers with non-textual features. In *SIGIR 06: Proceedings of the 29th Annual International ACM*, pages 228–235. ACM Press, 2006.

[19] Wei Li and Andrew Mccallum. Pachinko allocation: Dag-structured mixture models of topic correlations. In *In Proceedings of the 23rd International Conference on Machine Learning*, pages 577–584, 2006.

[20] Mingrong Liu, Yicen Liu, and Qing Yang. Predicting best answerers for new questions in community question answering. In *Proceedings of the 11th International Conference on Web-age Information Management*, pages 127–138. Springer-Verlag, 2010.

[21] Xiaoyong Liu, W. Bruce Croft, and Matthew B. Koll. Finding experts in community-based question-answering services. In *Proceedings of the 14th ACM Conference on Information and Knowledge Management*, pages 315–316, 2005.

[22] Andrew Kachites McCallum. Mallet: A machine learning for language toolkit. http://mallet.cs.umass.edu, 2002.

[23] David Mimno and Andrew Mccallum. Expertise modeling for matching papers with reviewers. In *Proceedings of the 13th ACM SIGKDD*, pages 500–509, 2007.

[24] Audris Mockus and James D. Herbsleb. Expertise browser: A quantitative approach to identifying expertise. In *Proceedings of International Conference on Software Engineering (ICSE 2002)*, pages 503–512, 2002.

[25] Maria Soledad Pera and Yiu-Kai Ng. A community question-answering refinement system. In *Proceedings of the 22nd ACM Conference on Hypertext and Hypermedia*, pages 251–260. ACM, 2011.

[26] Desislava Petkova and W. Bruce Croft. Hierarchical language models for expert finding in enterprise corpora. *Int. J. on AI Tools*, 2008.

[27] Jim Pitman and Marc Yor. The two-parameter poisson-dirichlet distribution derived from a stable subordinator. *The Annals of Probability*, 25:855–900, 1997.

[28] Jay M. Ponte and W. Bruce Croft. A language modeling approach to information retrieval. In *Proceedings of the 21th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '98, pages 275–281. ACM, 1998.

[29] Mingcheng Qu, Guang Qiu, Xiaofei He, Cheng Zhang, Hao Wu, Jiajun Bu, and Chun Chen. Probabilistic question recommendation for question answering communities. In *Proceedings of the 18th International Conference on World Wide web*, WWW '09, pages 1229–1230. ACM, 2009.

[30] Michal Rosen-Zvi, Thomas Griffiths, Mark Steyvers, and Padhraic Smyth. The author-topic model for authors and documents. In *Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence*, pages 487–494. AUAI Press, 2004.

[31] Gerard Salton and Christopher Buckley. Term-weighting approaches in automatic text retrieval. In *Information Processing and Management*, pages 513–523, 1988.

[32] M. Mahdi Shafiei and Evangelos E. Milios. Latent Dirichlet co-clustering. In *ICDM '06: Proceedings of the Sixth International Conference on Data Mining*, pages 542–551, Washington, DC, USA, 2006. IEEE Computer Society.

[33] Yee Whye Teh, Michael I. Jordan, Matthew J. Beal, and David M. Blei. Hierarchical dirichlet processes. *Journal of the American Statistical Association*, 101:1566–1581, 2006.

[34] Yee Whye Teh, Michael I. Jordan, Matthew J. Beal, and David M. Blei. Hierarchical dirichlet processes. *Journal of the American Statistical Association*, 101(476):1566–1581, 2006.

[35] Xing Wei and W. Bruce Croft. Lda-based document models for ad-hoc retrieval. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '06, pages 178–185, New York, NY, USA. ACM.

[36] Chengxiang Zhai and John Lafferty. A study of smoothing methods for language models applied to information retrieval. *ACM Trans. Inf. Syst.*, 22:179–214, 2004.

Appendix A

First Appendix

The details of Gibbs sampling derivations for the UPM model is presented in this section.

We begin with the joint distribution $P(w, Z, Y|\alpha_\pi, \alpha_\phi, \alpha_\beta)$ that according to the Bayes rules, the formula can be simplified.

$$\begin{aligned} P(w, Z, Y|\alpha_\pi, \alpha_\phi, \alpha_\beta) &= \iiint P(w, Z, Y, \pi, \phi, \beta|\alpha_\pi, \alpha_\phi, \alpha_\beta)\mathrm{d}\pi\mathrm{d}\phi\mathrm{d}\beta \\ &= \int P(w|Z, \beta)P(\beta|\alpha_\beta\mathrm{d}\beta) \int P(Z|Y, \phi)P(\phi|\alpha_\phi\mathrm{d}\phi) \int P(Y|\pi)P(\pi|\alpha_\pi\mathrm{d}\pi) \end{aligned} \tag{A.1}$$

Each component can be computed as follows:

$$\begin{aligned} &\int P(w|Z, \beta)P(\beta|\alpha_\beta) = \int \prod_{u=1}^{U}\prod_{q=1}^{N_Q}\prod_{n=1}^{N_{WQ}} P(w_{qn}^u|w_{qn}^u, \beta)P(\beta|\alpha_\beta)\mathrm{d}\beta \\ &= \int \prod_{k=1}^{K}\prod_{v=1}^{VQ}[\beta^{kv}]^{C_q^{kv}} \prod_{k=1}^{K}\frac{\Gamma VQ \times \alpha_\beta}{\Gamma{\alpha_\beta}^{VQ}} \prod_{v=1}^{VQ}[\beta^{kv}]^{\alpha_\beta - 1}\mathrm{d}\beta \\ &= \prod_{k=1}^{K}\frac{\Gamma VQ \times \alpha_\beta}{\Gamma\alpha_\beta^{VQ}} \int \prod_{k=1}^{K}\prod_{v=1}^{VQ}[\beta^{kv}]^{C_q^{kv}+\alpha_\beta-1}\mathrm{d}\beta \\ &= \prod_{k=1}^{K}\frac{\Gamma VQ \times \alpha_\beta}{\Gamma\alpha_\beta^{VQ}} \frac{\prod_{v=1}^{VQ}\Gamma\alpha_\beta + C_q^{kv}}{\Gamma[\sum_{v=1}^{VQ}(\alpha_\beta + C_q^{kv})]} \end{aligned} \tag{A.2}$$

$$\begin{aligned} &\int P(Z|Y, \phi)P(\phi|\alpha_\phi) = \int \prod_{u=1}^{U}\prod_{q=1}^{N_Q}\prod_{n=1}^{N_{WQ}} P(Z_{qn}^u|Y_q^u, \phi)P(\phi|\alpha_\phi)\mathrm{d}\phi \\ &= \int \prod_{p=1}^{P}\prod_{k=1}^{K}[\phi^{pk}]^{C_u^{pk}} \prod_{p=1}^{P}\frac{\Gamma K \times \alpha_\phi}{\Gamma{\alpha_\phi}^{K}} \prod_{k=1}^{K}[\phi^{pk}]^{\alpha_\phi - 1}\mathrm{d}\phi \\ &= \prod_{p=1}^{P}\frac{\Gamma K \times \alpha_\phi}{\Gamma\alpha_\phi^{K}} \int \prod_{p=1}^{P}\prod_{k=1}^{K}[\phi^{pk}]^{C_u^{pk}+\alpha_\phi-1}\mathrm{d}\phi \\ &= \prod_{p=1}^{P}\frac{\Gamma K \times \alpha_\phi}{\Gamma\alpha_\phi^{K}} \frac{\prod_{k=1}^{K}\Gamma\alpha_\phi + C_u^{pk}}{\Gamma[\sum_{k=1}^{K}(\alpha_\phi + C_u^{pk})]} \end{aligned} \tag{A.3}$$

$$
\begin{aligned}
&\int P(Y|Y,\pi)P(\pi|\alpha_\pi) = \int \prod_{u=1}^{U}\prod_{q=1}^{NQ} P(Y_q^u|\pi)P(\pi|\alpha_\pi)\mathrm{d}\pi \\
&= \int \prod_{u=1}^{U}\prod_{p=1}^{P}[\pi^u]^{C_u^p}\prod_{u=1}^{U}\frac{\Gamma P\times\alpha_\pi}{\Gamma {\alpha_\pi}^P}\prod_{p=1}^{P}[\pi^p]^{\alpha_\pi-1}\mathrm{d}\pi \\
&= \prod_{u=1}^{U}\frac{\Gamma P\times\alpha_\pi}{\Gamma\alpha_\pi^P}\int \prod_{u=1}^{U}\prod_{p=1}^{P}[\pi^u]^{C_u^p+\alpha_\pi-1}\mathrm{d}\pi \\
&= \prod_{u=1}^{U}\frac{\Gamma P\times\alpha_\pi}{\Gamma\alpha_\pi^P}\frac{\prod_{p=1}^{P}\Gamma\alpha_\pi+C_u^p}{\Gamma[\sum_{p=1}^{P}(\alpha_\pi+C_u^p)]}
\end{aligned}
\tag{A.4}
$$

We use chain rule for inference derivations, we have:

$$
\begin{aligned}
&P(w,Z,Y|\alpha_\pi,\alpha_\beta,\alpha_\phi) \\
&= \prod_{k=1}^{K}\frac{\Gamma(VQ\alpha_\beta)}{\Gamma(\alpha_\beta)^{VQ}}\times\frac{\prod_{v=1}^{VQ}\Gamma(\alpha_\beta+C^{kv})}{\Gamma(\sum_{v=1}^{VQ}(\alpha_\beta+C^{kv}))} \\
&\times\prod_{p=1}^{P}\frac{\Gamma(K\alpha_\beta)}{\Gamma(\alpha_\beta)^{K}}\times\frac{\prod_{k=1}^{K}\Gamma(\alpha_\phi+C^{pk})}{\Gamma(\sum_{k=1}^{K}(\alpha_\phi+C^{pk}))} \\
&\times\prod_{u=1}^{U}\frac{\Gamma(P\alpha_\pi)}{\Gamma(\alpha_\pi)^{P}}\times\frac{\prod_{p=1}^{P}\Gamma(\alpha_\pi+C^{up})}{\Gamma(\sum_{p=1}^{P}(\alpha_\pi+C^{up}))} \\
&= P(W|Z)P(Z|Y)P(Y)
\end{aligned}
\tag{A.5}
$$

$$
\begin{aligned}
&P(Z_{qn}^u=k|w_{q\overline{n}}^u,Y_q^u,Z_{q\overline{n}}^u) = \frac{P(Y_{qn}^u,Y_q^u,w_{qn}^u)}{P(Y_{q\overline{n}}^u,Y_q^u,w_{q\overline{n}}^u)} \\
&= \frac{\alpha_\beta+C_{qu}^{kw_{qu}^n}}{\sum_{v=1}^{V}(\alpha_\beta+C_{qu}^{kv})}\times\frac{\alpha_\phi+C_u^{pk}}{\sum_{k=1}^{K}\alpha_{phi}+C_u^{pk}}
\end{aligned}
\tag{A.6}
$$

$$
\begin{aligned}
&P(Y_u^q=p|w,Z,Y_{\overline{q}}^u) = \frac{P(w,Z,Y|\alpha_\pi,\alpha_\pi,\alpha_\beta)}{P(w,Z,Y_{\overline{q}}^u|\alpha_\pi,\alpha_\pi,\alpha_\beta)} \\
&= \frac{P(w|Z,\alpha_\beta)P(Z|Y,\alpha_\phi)P(Y|\alpha_\pi)}{P(w|Z,\alpha_\beta)P(Z_{\overline{q}^u}|Y_{\overline{q}^u},\alpha_\phi)P(Z_q^u)P(Y_q^u|\alpha_\pi)} \\
&= \frac{\prod_{n=1}^{N}(\alpha_\beta+C_{qu}^{kwn})}{\prod_{n=1}^{N}(\sum_{v=1}^{VQ}(\alpha_\beta)+C_{qu}^{kv})}\times\frac{\prod_{n=1}^{N}(\alpha_\phi+C_u^{pk})}{\prod_{n=1}^{N}(\sum_{k=1}^{K}(\alpha_\phi)+C_u^{pk})}\times\frac{\alpha_\pi+C_u^p}{\sum_{p=1}^{P}(\alpha_\pi+C_u^p)}
\end{aligned}
\tag{A.7}
$$

Table A.1: A brief description of symbols used for the UPM model derivations

Notation	Description
VQ	Total number of words in vocabulary
C_q^{kv}	Total number of times topic k is assigned to word v
C_u^{pk}	Total number of times persona p is assigned to topic k
C_u^p	Total number of times persona p is assigned to user u
$w_{q\overline{n}}^u$	All the words in the vocabulary excluding word n
$Z_{q\overline{n}}^u$	Excluding the topic assignment for word n
$Y_{\overline{q}}^u$	Excluding the persona assignment for question q

Appendix B

Two-parameter Poisson Dirichlet process

The two parameter Poisson-Dirichlet process (PDP), is defined as generalization of Dirichlet process. In the STM model, ν is a distribution over topics of a question. a PDP prior is considered on $\nu : \nu \sim PDP(a, b, \mu)$, where μ is a base distribution; $a(0 \leq a \leq 1)$ and $b(b > -a)$. b, is a strength parameter that control the amount of variability around μ.
PDP distribution is used to define the relation between μ_u and $\nu_{u,q}$ as $\nu_{u,q} \sim PDP(a, b, \nu_{u,q})$. However, a distribution such as $\nu_{u,q} \sim Dirichlet(b\mu_i)$ where b is the equivalent sample size. The problem of the later distribution is that parameter vectors such as μ_u cannot be integrated out. Therefore, the following lemma has been used [11]:

$$\begin{aligned} PDP(0, b, discrete(\theta)) &\approx Dirichlet(b\theta) \\ PDP(a, 0, discrete(\theta)) &\approx Dirichlet(a\theta)(a \to 0) \end{aligned} \tag{B.1}$$

Appendix C

Computational Complexity

Topic models are known as complex models. However, there is some research that shows the complexity of LDA and K-means are comparable [35]. They estimated the running time for each iteration of K-means and showed that it grows linearly with the number of documents (N) and the number of classes (K), i.e., $O(KN)$. They also demonstrated that the complexity of the LDA's Gibbs sampling in each iteration is linear with the number of topics and the number of documents ($O(KN)$).

The Gibbs sampling is the time-consuming part of the topic models. In the LDA model, this part is linear with the number of iteration (I), the number of topics (K), the number of documents (N) and the average number of words in each document (N_w). The STM model, on the other hand, is more complex than LDA as it has another high level topic selection. Its complexity is linear with the number of iteration, the number of high level topics, the number of low level topics, the number of documents and the average number of words in each document.

Similar to the complexity, the high level topics effects the memory usage of the STM model compared to the LDA model. LDA basically needs two matrix one for the topic assignments of documents, and the other one for the word distribution of each topic. STM, on the other hand, has an additional matrix for the topic distribution of each segments.